EViews 5.1 User's Guide

Quantitative Micro Software

EViews 5.1 User's Guide

Copyright © 1994–2005 Quantitative Micro Software, LLC
All Rights Reserved
Printed in the United States of America

This software product, including program code and manual, is copyrighted, and all rights are reserved by Quantitative Micro Software, LLC. The distribution and sale of this product are intended for the use of the original purchaser only. Except as permitted under the United States Copyright Act of 1976, no part of this product may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of Quantitative Micro Software.

Disclaimer

The authors and Quantitative Micro Software assume no responsibility for any errors that may appear in this manual or the EViews program. The user assumes all responsibility for the selection of the program to achieve intended results, and for the installation, use, and results obtained from the program.

Trademarks

Windows, Windows 95/98/2000/NT/Me/XP, and Microsoft Excel are trademarks of Microsoft Corporation. PostScript is a trademark of Adobe Corporation. X11.2 and X12-ARIMA Version 0.2.7 are seasonal adjustment programs developed by the U. S. Census Bureau. Tramo/Seats is copyright by Agustin Maravall and Victor Gomez. All other product names mentioned in this manual may be trademarks or registered trademarks of their respective companies.

Quantitative Micro Software, LLC
4521 Campus Drive, #336, Irvine CA, 92612-2621
Telephone: (949) 856-3368
Fax: (949) 856-2044
e-mail: sales@eviews.com
web: www.eviews.com
March 17, 2005

Table of Contents

WHAT'S NEW IN EVIEWS 5.0 1
What's New in 5.0 1
Compatibility Notes 4
EVIEWS 5.1 UPDATE OVERVIEW 9
Overview of EViews 5.1 New Features 9
PREFACE 13
PART I. EVIEWS FUNDAMENTALS 15
CHAPTER 1. INTRODUCTION 17
What is EViews? 17
Installing and Running EViews 17
Windows Basics 18
The EViews Window 21
Closing EViews 25
Where to Go For Help 25
CHAPTER 2. A DEMONSTRATION 27
Getting Data into EViews 27
Examining the Data 30
Estimating a Regression Model 36
Specification and Hypothesis Tests 38
Modifying the Equation 40
Forecasting from an Estimated Equation 42
Additional Testing 46
CHAPTER 3. WORKFILE BASICS 49
What is a Workfile? 49
Creating a Workfile 50
The Workfile Window 57
Saving a Workfile 61
Loading a Workfile 63
Multi-page Workfiles 64
Addendum: File Dialog Features 71

CHAPTER 4. OBJECT BASICS 73
What is an Object? 73
Basic Object Operations 76
The Object Window 78
Working with Objects 81
CHAPTER 5. BASIC DATA HANDLING 87
Data Objects 87
Samples 95
Sample Objects 103
Importing Data 105
Exporting Data 114
Frequency Conversion 115
Importing ASCII Text Files 120
CHAPTER 6. WORKING WITH DATA 129
Numeric Expressions 129
Series 137
Auto-series 141
Groups 145
Scalars 148
CHAPTER 7. WORKING WITH DATA (ADVANCED) 149
Auto-Updating Series 149
Alpha Series 153
Date Series 160
Value Maps 163
CHAPTER 8. SERIES LINKS 177
Basic Link Concepts 177
Creating a Link 191
Working with Links 201
CHAPTER 9. ADVANCED WORKFILES 207
Structuring a Workfile 207
Resizing a Workfile 231
Appending to a Workfile 234
Contracting a Workfile 237

Copying from a Workfile . . . 238
Reshaping a Workfile . . . 241
Sorting a Workfile . . . 259
Exporting from a Workfile . . . 259

CHAPTER 10. EVIEWS DATABASES . . . 261
Database Overview . . . 261
Database Basics . . . 262
Working with Objects in Databases . . . 267
Database Auto-Series . . . 274
The Database Registry . . . 275
Querying the Database . . . 277
Object Aliases and Illegal Names . . . 285
Maintaining the Database . . . 287
Foreign Format Databases . . . 289
Working with DRIPro Links . . . 300

PART II. BASIC DATA ANALYSIS . . . 307

CHAPTER 11. SERIES . . . 309
Series Views Overview . . . 309
Spreadsheet and Graph Views . . . 309
Descriptive Statistics . . . 310
Tests for Descriptive Stats . . . 315
Distribution Graphs . . . 322
One-Way Tabulation . . . 325
Correlogram . . . 326
Unit Root Test . . . 329
BDS Test . . . 329
Properties . . . 332
Label . . . 333
Series Procs Overview . . . 334
Generate by Equation . . . 334
Resample . . . 334
Seasonal Adjustment . . . 336
Exponential Smoothing . . . 351
Hodrick-Prescott Filter . . . 356
Frequency (Band-Pass) Filter . . . 358

CHAPTER 12. GROUPS 363
Group Views Overview 363
Group Members 363
Spreadsheet 363
Dated Data Table 365
Graphs 374
Multiple Graphs 376
Descriptive Statistics 379
Tests of Equality 380
N-Way Tabulation 381
Principal Components 385
Correlations, Covariances, and Correlograms 386
Cross Correlations and Correlograms 387
Cointegration Test 387
Unit Root Test 387
Granger Causality 388
Label 389
Group Procedures Overview 389
CHAPTER 13. STATISTICAL GRAPHS FROM SERIES AND GROUPS 391
Distribution Graphs of Series 391
Scatter Diagrams with Fit Lines 400
Boxplots 409
CHAPTER 14. GRAPHS, TABLES, AND TEXT OBJECTS 415
Creating Graphs 415
Modifying Graphs 416
Multiple Graphs 425
Printing Graphs 427
Copying Graphs to the Clipboard 428
Saving Graphs to a File 429
Graph Commands 429
Creating Tables 429
Table Basics 430
Basic Table Customization 432
Customizing Table Cells 433
Copying Tables to the Clipboard 437

Saving Tables to a File 438
Table Commands 438
Text Objects 438

PART III. BASIC SINGLE EQUATION ANALYSIS 441

CHAPTER 15. BASIC REGRESSION 443
Equation Objects 443
Specifying an Equation in EViews 444
Estimating an Equation in EViews 447
Equation Output 449
Working with Equations 455
Estimation Problems 460

CHAPTER 16. ADDITIONAL REGRESSION METHODS 461
Special Equation Terms 461
Weighted Least Squares 469
Heteroskedasticity and Autocorrelation Consistent Covariances 471
Two-stage Least Squares 473
Nonlinear Least Squares 480
Generalized Method of Moments (GMM) 488

CHAPTER 17. TIME SERIES REGRESSION 493
Serial Correlation Theory 493
Testing for Serial Correlation 494
Estimating AR Models 497
ARIMA Theory 501
Estimating ARIMA Models 503
ARMA Equation Diagnostics 512
Nonstationary Time Series 517
Unit Root Tests 518
Panel Unit Root Tests 530

CHAPTER 18. FORECASTING FROM AN EQUATION 543
Forecasting from Equations in EViews 543
An Illustration 546
Forecast Basics 549
Forecasts with Lagged Dependent Variables 555
Forecasting with ARMA Errors 557

Forecasting from Equations with Expressions 561
Forecasting with Expression and PDL Specifications 567

CHAPTER 19. SPECIFICATION AND DIAGNOSTIC TESTS 569
Background 569
Coefficient Tests 570
Residual Tests 579
Specification and Stability Tests 584
Applications 593

PART IV. ADVANCED SINGLE EQUATION ANALYSIS 599

CHAPTER 20. ARCH AND GARCH ESTIMATION 601
Basic ARCH Specifications 601
Estimating ARCH Models in EViews 604
Working with ARCH Models 610
Additional ARCH Models 612
Examples 617

CHAPTER 21. DISCRETE AND LIMITED DEPENDENT VARIABLE MODELS 621
Binary Dependent Variable Models 621
Estimating Binary Models in EViews 623
Procedures for Binary Equations 633
Ordered Dependent Variable Models 638
Estimating Ordered Models in EViews 639
Views of Ordered Equations 642
Procedures for Ordered Equations 643
Censored Regression Models 644
Estimating Censored Models in EViews 645
Procedures for Censored Equations 650
Truncated Regression Models 654
Procedures for Truncated Equations 655
Count Models 658
Views of Count Models 662
Procedures for Count Models 662
Demonstrations 663
Technical Notes 667

CHAPTER 22. THE LOG LIKELIHOOD (LOGL) OBJECT 671
Overview 671
Specification 673
Estimation 679
LogL Views 680
LogL Procs 681
Troubleshooting 684
Limitations 685
Examples 687
PART V. MULTIPLE EQUATION ANALYSIS 693
CHAPTER 23. SYSTEM ESTIMATION 695
Background 695
System Estimation Methods 696
How to Create and Specify a System 698
Working With Systems 706
Technical Discussion 712
CHAPTER 24. VECTOR AUTOREGRESSION AND ERROR CORRECTION MODELS 721
Vector Autoregressions (VARs) 721
Estimating a VAR in EViews 722
VAR Estimation Output 723
Views and Procs of a VAR 724
Structural (Identified) VARs 733
Cointegration Test 739
Vector Error Correction (VEC) Models 749
A Note on Version Compatibility 752
CHAPTER 25. STATE SPACE MODELS AND THE KALMAN FILTER 753
Background 753
Specifying a State Space Model in EViews 758
Working with the State Space 769
Converting from Version 3 Sspace 775
Technical Discussion 775
CHAPTER 26. MODELS 777
Overview 777

An Example Model 780
Building a Model 794
Working with the Model Structure 796
Specifying Scenarios 800
Using Add Factors 802
Solving the Model 804
Working with the Model Data 817

PART VI. PANEL AND POOLED DATA 823

CHAPTER 27. POOLED TIME SERIES, CROSS-SECTION DATA 825
The Pool Workfile 825
The Pool Object 826
Pooled Data 829
Setting up a Pool Workfile 831
Working with Pooled Data 838
Pooled Estimation 845

CHAPTER 28. WORKING WITH PANEL DATA 873
Structuring a Panel Workfile 873
Panel Workfile Display 875
Panel Workfile Information 877
Working with Panel Data 881
Basic Panel Analysis 893

CHAPTER 29. PANEL ESTIMATION 901
Estimating a Panel Equation 901
Panel Estimation Examples 908
Panel Equation Testing 922
Estimation Background 930

APPENDIX A. GLOBAL OPTIONS 937
The Options Menu 937
Print Setup 943

APPENDIX B. WILDCARDS 945
Wildcard Expressions 945
Using Wildcard Expressions 945
Source and Destination Patterns 946

Resolving Ambiguities 947
Wildcard versus Pool Identifier 948

APPENDIX C. ESTIMATION AND SOLUTION OPTIONS 951
Setting Estimation Options 951
Optimization Algorithms 956
Nonlinear Equation Solution Methods 959

APPENDIX D. GRADIENTS AND DERIVATIVES 963
Gradients 963
Derivatives 966

APPENDIX E. INFORMATION CRITERIA 971
Definitions 971
Using Information Criteria as a Guide to Model Selection 972

REFERENCES 973

INDEX 983

What's New in EViews 5.0

EViews 5.0 features the most extensive changes and improvements since the initial release of EViews in 1994. New data structures and objects provide you with powerful new tools for working with data, while new graphics and table support give you additional control over the display of information. Other improvements include powerful new estimation techniques and new methods of working with samples.

What's New in 5.0

The following is an abbreviated list of the major new features of EViews 5.0:

Workfiles

- Multi-page workfiles.
- Support for complex data structures including irregular dated data, cross-section data with observation identifiers, dated and undated panel data.
- Merge, append, subset, resize, sort, and reshape (stack and unstack) workfiles.
- Data translation tools allow you to read from and write to various spreadsheet, statistical, and database formats: Microsoft Access files, Gauss Dataset files, ODBC Dsn files, ODBC Query files, SAS Transport files, native SPSS files, SPSS Portable files, Stata files, Excel files, raw ASCII text or binary files, HTML, or ODBC Databases and queries.

General Data

- Alphanumeric (string) series, with an extensive library of string manipulation functions.
- Date series, with extensive library of date manipulation functions.
- Dynamic frequency conversion and match merging using link objects. Frequency conversion and match merge links will be updated whenever the underlying data change.
- Auto-updating series that depend upon a formula are automatically recalculated whenever the underlying data change.
- Value labels (e.g., the labels "High", "Med", "Low", corresponding to the values 2, 1, 0) may be used with numeric and alpha series. Function support allows you to work with either the underlying or the mapped values.

- Improved sample object processing allows for the direct use of sample objects in series expressions. In addition, sample objects may now be used with set operators, allowing you to create sample objects from existing sample objects using the operators "AND", "OR", and "NOT".
- New family of by-group statistics facilitates assigning to observations the values from the computation of subgroup statistics.
- Automatic creation of sets of dummy variables for use in estimation.

String Support

- Full support for strings and string variables.
- New library of string functions and operators.
- Functions for converting between date values and string representations of dates.

Date Support

- Full support for calendar dates with extensive library of functions for manipulating dates and date values.
- Functions for converting between date values and string or numeric representations of dates.

Panel and Pooled Data

General

- Workfile tools for **reshaping data to and from panel (stacked) and pool (unstacked) workfile structures**.
- **Panel unit root tests** (Levin-Lin-Chu, Breitung, Im-Pesaran-Shin, Fisher-type tests using ADF and PP tests—Maddala-Wu and Choi, Hadri).
- Linear equation estimation with additive cross-section and period effects (fixed or random). Random effects models available in linear specifications only. Two-way random and mixed effects models supported for balanced linear data only, all others for both balanced and unbalanced data.
- Quadratic unbiased estimators (QUEs) for component variances in random effects models (Swamy-Arora, Wallace-Hussain, Wansbeek-Kapteyn).
- Generalized least squares for models with cross-section or period heteroskedastic and correlated specifications. Support for both one-step and iterative weighting.
- Two-stage least squares (2SLS) / Instrumental variables (IV) estimation with cross-section and period fixed or random effects. Generalized 2SLS/IV estimation of GLS specifications.

- Most specifications support estimation with AR errors using nonlinear least squares on the transformed specification.
- Robust standard error calculations including seven types of White and Panel-corrected standard errors (PCSE).

Panel Specific

- Structured workfiles support large cross-section panels.
- Panel data graphs. Various plots by cross-section in multiple graphs or combined. Graphs of summary values across cross-section.
- Nonlinear estimation with additive effects.
- GMM estimation for models with cross-section or period heteroskedastic and correlated specifications. Support for both one-step and iterative weighting.
- Linear dynamic panel data estimation using first differences or orthogonal deviations and period specific instruments (Arellano-Bond one-step, one-step robust, two-step, iterated). Flexible specification of instrument list.

Pool Specific

- Define groups of cross-sections for dummy variable processing.
- Support for period specific coefficients and instruments.

GARCH Estimation

- Student's t and Generalized Error Distribution GARCH estimation with optional fixed distribution parameter.
- More flexible EGARCH and TARCH specifications allow for estimation of a wider range of econometric models.
- Power ARCH specifications with optional fixed power parameter.

Other Statistical and Econometric

- Confidence ellipses showing the joint confidence region of any two functions of estimated parameters from an EViews estimation object.
- ARMA equation diagnostics. Display the inverse roots of the AR and/or MA characteristic polynomial; compare the theoretical (estimated) autocorrelation pattern with the actual correlation pattern for the structure residuals; display the ARMA impulse response to an innovation shock.
- Band-pass (frequency) filters for a series object. EViews currently computes the Baxter-King, Christiano-Fitzgerald fixed length, and the Christiano-Fitzgerald asymmetric full sample filters.

Graphs and Tables

- Filled area graphs.
- Boxplots.
- Enhanced table customization, with control over font face, font size and color, cell background color, and borders, with cell merging and annotation.
- Improved interactive and command interface for working with tables. Selecting cells, resizing columns, and changing numeric and other display formats should be much more straightforward and intuitive.
- Enhanced graph output. Write graphs as PostScript files. Improved Windows Metafile support now comes with control over output sizing.
- Tables may be written to HTML and RTF files.

General

- Improved speed of operation.

Compatibility Notes

Relational Comparisons with Missing Values

The behavior of relational operators with missing values has changed in EViews 5. Unlike previous versions, equality (" = ") and inequality (" < > ") comparisons of series and matrix objects involving NA values propagate the NA values. Note that this behavior differs from previous versions of EViews where NAs were treated as ordinary values for purposes of these comparisons. The change in behavior was necessary to support the use of string missing values.

There is one special case where these comparisons have not changed. If you test equality or inequality against a literal NA value (*e.g.*, "X = NA") in Version 4 or 5, the literal is treated as an ordinary value for the purpose of equality and inequality comparison.

You may obtain the Version 4 behavior using the special functions @EQNA and @NEQNA to perform equality and strict inequality comparisons without propagating NAs. In addition, programs may be run in version 4 compatibility mode to enable the earlier behavior of comparisons for element operations. Note that compatibility mode does not apply to string comparisons that assign values into EViews numeric or alpha series. See "Comparisons Involving NAs/Missing Values" on page 96 of the *Command and Programming Reference* for additional detail.

See also "Missing Values" on page 134 for additional discussion.

String and Replacement Variable Substitution

There are two important changes in the way EViews 5 handles string and replacement variable substitution. First, the use of contextual information to distinguish between the use of string and replacement variables has been eliminated. Second, text which is potentially a string variable is no longer substituted for when used inside of a string expression.

To address compatibility issues in existing programs, EViews 5 provides a compatibility mode so that programs may be run using the version 4 substitution rules.

See "Version 5 Compatibility Notes" on page 93 of the *Command and Programming Reference* for a detailed discussion.

Case-sensitive String Comparisons

In previous versions of EViews, batch program statements could involve string comparisons. In these settings, the comparisons were performed caselessly. Version 5 string comparisons are now case-sensitive.

Programs may be run in version 4 compatibility mode to enable caseless comparisons. Note that compatibility mode does not apply to string comparisons that assign values into EViews numeric or alpha series.

See "Case-Sensitive String Comparison" on page 96 of the *Command and Programming Reference* for additional detail.

Workfile Compatibility

With some exceptions, EViews 5 workfiles are backward compatible with EViews 4:

- Multi-page workfiles are not fully compatible with earlier versions of EViews since previous versions will only read the first page in a multi-page workfile. To ensure backward compatibility, you may save individual workfile pages as separate workfiles.
- Workfiles which are saved in compressed format are not backward compatible and cannot not be read by earlier versions of EViews.

In addition, the following objects are new, or have been modified in version 5, so that transporting them back into version 4 may result in data loss:

- Tables.
- Pools.
- Equations. Equations that employ new features (some ARCH equation estimators, panel equations) are not backward compatible.
- Valmaps.

- Alphas.
- Links.

If you save workfiles with tables, newly estimated pools, panel, and some ARCH equations, and attempt to read them in EViews 4.1 or earlier, EViews will delete the incompatible object, and notify you of the deletion. To prevent loss, we recommend that you make a copy of any workfiles that contain these objects if you would like to use these workfiles in both version 4 and 5 of EViews.

Miscellaneous Issues

- The default seeding of the pseudo-random number generators has changed. See `rndseed` (p. 425) in the *Command and Programming Reference* for details.
- We have updated some of our matrix computational routines from Linpack/Eispack to Lapack. A consequence of this change is that the eigenvectors (which are not uniquely identified) may appear to be different from version 4 and, in particular, the signs may flip. The routines that are most likely to be affected by this change are the principal components and singular value decomposition.
- The critical values displayed in the cointegration test output were based on Osterwald-Lenum (1992) up to EViews 4. These have been replaced by the more accurate ones based on the response surface coefficients of MacKinnon-Haug-Michelis (1999); see `coint` (p. 245) in the *Command and Programming Reference* for details.
- The system log likelihood statistics reported at the bottom of the VAR/VEC estimation output have been changed. In version 4, the log likelihood and information criteria were based on an estimate of the residual covariance matrix that was adjusted for degrees of freedom (for the estimated parameters). In version 5, these statistics are now based on the maximum likelihood estimate of the innovation covariance matrix which does *not* adjust for degrees of freedom.
- Previously, the standardized residual views and procedures of equations only adjusted for models with prior weighting (weighted least squares) and models with time-varying variances (ARCH, GARCH, *etc.*). We have modified the behavior of these routines so that standardization yields residuals which are always divided by the estimate of the residual standard deviation (03/15/2005).
- The definition of the R-squared statistic in weighted least squares models has been modified to align with the calculation of the F-statistic test statistic of the significance of the non-constant regressors. The F-statistic and test based on this statistic have not been changed. In addition, the residual based tests for ARCH, serial correlation, and heteroskedasticity in weighted least squares models have been updated to use the new calculations. You may find that your results for these latter tests in

weighted least squares models differ from previous versions of EVIews (03/15/2005).

EViews 5.1 Update Overview

We are very pleased to offer you a free upgrade from EViews 5.0 to EViews 5.1. The upgrade, which provides several new features and improvements to the existing program, is in part a response to user requests, and is in part a collection of features that were not completed in time for the release of EViews 5.0.

Some of these features, like the enhanced graph customization tools and Enterprise Edition support for EcoWin online data, represent significant improvements in the set of tools for working with your data. Other features, such as improved support for creating workfiles from identifiers, an enhanced `copy` command, and expanded testing in panel and pool estimation, are minor improvements to existing routines.

Overview of EViews 5.1 New Features

Graph Customization

EViews 5.1 features a greatly expanded set of tools for customizing graphics. These tools allow for added control over the graph area, frame, and background, font characteristics, axes, grid lines, and more. New default templates provide easy-to-use examples of graph customization, and can be used as the basis for user template creation.

Graph Appearance Settings

EViews 5.1 provides control over a wider range of graph appearance characteristics.

- You may now select graph background color, fill color, frame color, and frame thickness.
- Graph legends, and text object boxes, now feature user-specified fill color, frame color, and frame thickness.
- Font face, text color, and text style (bold, italic, underline, strikeout) choice has been extended to all text in graphs, including legend, axes, and text objects.
- Axes options allow for any combination of axes to be displayed with user-specified line widths.
- Grid line characteristics (color, width, pattern) may now be specified by the user.
- The data portion of a graph may now be indented horizontally and/or vertically within the graph frame using user-selected increments.

- The background color in a graph may now be printed and exported with the graph.

Note that full command line support has been provided for all of the above features.

Improved Graph Template Support

Graph templates have been improved so that they control a wider range of appearance settings and are easier to use:

- Template support is provided for all of the new appearance settings.
- EViews now provides a set of predefined templates illustrating the use of basic display options such as background color, fill color, line/bar coloring, graph size, and grid line settings. The predefined templates, which include "Classic" (classic EViews), "Modern", "Reverse", "Midnight", "Spartan", and "Monochrome", may be used as-is to modify the appearance of graphs, or may serve as the basis for further customization or template creation.
- Templates may now be applied to an existing graph via the main graph dialog, and may also be used to update graph defaults.

Enhanced Graphics Defaults

Major improvements in the setting and handling of graph defaults allow you to control the appearance of newly created graphs:

- Global defaults have been extended to support all of the new appearance settings.
- Both global and individual graph object defaults may be updated using templates.
- Individual defaults allow you to specify the settings for new text, line, and shade objects added to an existing graph object. For text, you may specify font options (font name, size, style, and color), and the fill and frame color to be used when the text is enclosed in a box. For lines or shades, you may specify color, width, and pattern.

Simply customize the appearance of a single graph, then instantly update your defaults so that new graphs will take on the desired appearance. Or, update the text and shade defaults in an existing graph, and subsequent new text and shade objects will use those settings.

Tools for Creating Workfile Pages from Identifiers

EViews 5 provides basic tools for creating a new workfile page from the unique values of one or more identifier series in a given workfile page. EViews 5.1 extends these tools to work with identifier series located in multiple pages, allowing you easily to create new pages structured using the values found in series or to form pages by crossing the unique

values from two identifier series, or by crossing the unique values from a single identifier series with a date frequency and range.

You may use these tools to create a new workfile page from: (1) the union of unique ID values from one more more pages; (2) the intersection of unique ID values from mulitple pages; (3) the cross of the unique values of two ID series, or (4) the cross of a single ID series with a date range.

Panel and Pool Equation Specification Testing

EViews 5.1 provides upgraded support for specification testing in panel or pool equations estimation. The following tests have been added in the EViews 5.1 upgrade:

- LR-type testing for omitted or redundant regressors in panel and pool equations specified by list. The omitted variable test enables you to add a set of variables to an existing panel or pool equation, and to ask whether the set makes a significant contribution to explaining the variation in the dependent variable. The redundant variables test allows you to test for the statistical significance of a subset of the variables included in your panel or pool equation.
- Redundant fixed effects testing for panel and pool equations estimated by ordinary linear and nonlinear least squares evaluates the statistical significance of the estimated fixed effects.
- Hausman random effects testing evaluates the restriction that the random effects are uncorrelated with the explanatory variables. The test statistic evaluates the closeness of the coefficients from a random effects pool equation to the corresponding fixed effects specification.

EcoWin Database Support

EViews 5.1 Enterprise Edition now supports direct access to on-line EcoWin databases (www.ecowin.com). The EcoWin Economic and Financial databases contain global international macroeconomic and financial data from more than 100 countries and multinational aggregates. Additional databases provide access to equities information and detailed country-specific information on earnings estimates, equities, funds, fixed income, and macroeconomics.

The EViews EcoWin interface provides the following features:

- An interactive graphical window attached to an EcoWin database allowing for browsing of series in the database, selection of series, and copying/exporting of series into an EViews workfile or another EViews database.
- A set of commands to perform tasks such as fetching a particular series by mnemonic from a selected EcoWin database. These tools may be used interactively or within EViews user-written programs.

EcoWin support is provided through the addition of an EcoWin database type to the list of databases supported by EViews. Since access to EcoWin data is provided using standard EViews database tools, most of the user interface to EcoWin data will be familiar to EViews users.

Miscellaneous

Other features added in EViews 5.1 include:

- The `copy` command has been enhanced, and now supports copying objects between named workiles and workfile pages. If copying series into a workfile page, EViews allows for automatic frequency conversion and match merging of the data in the series to the new workfile page frequency or structure.
- An equation forecast option allows you to ignore coefficient uncertainty when computing the forecast standard error.
- ARCH equations now allow you to calculate conditional variances as well as conditional standard deviations. In addition, you may now display and save the permanent component of the GARCH conditional variances in component models.
- A new global option allows you to set the default maximum number of errors in program execution.
- The `@cbvnorm` and `@dbvnorm` functions have been added, allowing you to evaluate the cumultative distribution function and density function of the standardized bivariate normal distribution.

Preface

This manual describes the interactive use of EViews, a program for statistical and econometric analysis, and forecasting. For details on the EViews command language, as well as a description of the programming and matrix languages, we refer you to the companion volume—the EViews *Command and Programming Reference*.

The manual is divided into six parts:

- Part I. "EViews Fundamentals" beginning on page 15—introduces you to the basics of using EViews. In addition to a discussion of basic Windows operations, we explain how to use EViews to manage your data.
- Part II. "Basic Data Analysis" beginning on page 307—describes the use of EViews to perform basic analysis of data and to draw graphs and create tables describing your data.
- Part III. "Basic Single Equation Analysis" on page 441—discusses standard regression analysis: ordinary least squares, weighted least squares, two-stage least squares, nonlinear least squares, time series analysis, specification testing and forecasting.
- Part IV. "Advanced Single Equation Analysis" beginning on page 599—documents autoregressive conditional heteroskedasticity (ARCH) models, discrete and limited dependent variable models, and user specified likelihood estimation.
- Part V. "Multiple Equation Analysis" on page 693—describes estimation and forecasting with systems of equations, vector autoregression and error correction models, state space models, pooled cross-section/time series data, and model solution.
- Part VI. "Panel and Pooled Data" on page 823—documents working with and estimating models with time series, cross-section data. The analysis may involve small numbers of cross-sections, with data unstacked series (pooled data) or large numbers systems of cross-sections, with stacked data (panel data).

You should not feel a need to read the manual from cover-to-cover in order to use EViews. We recommend, however, that you glance at most of Part I "EViews Fundamentals" to gain familiarity with the basic concepts and operation of the program. At a minimum, you should look over the first four chapters, especially the extended demonstration in Chapter 2, "A Demonstration", on page 27.

Part I. EViews Fundamentals

The following chapters document the fundamentals of working with EViews:

- Chapter 1, "Introduction" describes the basics of installing EViews.
- Chapter 2, "A Demonstration" guides you through a typical EViews session, introducing you to the basics of working with EViews.
- Chapter 3, "Workfile Basics" describes workfiles, the containers for your data in EViews.
- Chapter 4, "Object Basics" provides an overview of EViews objects, which are the building blocks for all analysis in EViews.
- Chapter 5, "Basic Data Handling" and Chapter 6, "Working with Data" provide background on the basics of working with numeric data. We describe methods of getting your data into EViews, manipulating and managing your data held in series and group objects, and exporting your data into spreadsheets, text files and other Windows applications.

We recommend that you browse through most of the material in the above section before beginning serious work with EViews.

The remaining material is somewhat more advanced:

- Chapter 7, "Working with Data (Advanced)", Chapter 8, "Series Links", and Chapter 9, "Advanced Workfiles" describe advanced tools for working with numeric data, and tools for working with different kinds of data (alphanumeric and date series, irregular and panel workfiles).
- Chapter 10, "EViews Databases" describes the EViews database features and advanced data handling features.

The advanced material is necessary only if you wish to work with the advanced tools.

Chapter 1. Introduction

What is EViews?

EViews provides sophisticated data analysis, regression, and forecasting tools on Windows-based computers. With EViews you can quickly develop a statistical relation from your data and then use the relation to forecast future values of the data. Areas where EViews can be useful include: scientific data analysis and evaluation, financial analysis, macroeconomic forecasting, simulation, sales forecasting, and cost analysis.

EViews is a new version of a set of tools for manipulating time series data originally developed in the Time Series Processor software for large computers. The immediate predecessor of EViews was MicroTSP, first released in 1981. Though EViews was developed by economists and most of its uses are in economics, there is nothing in its design that limits its usefulness to economic time series. Even quite large cross-section projects can be handled in EViews.

EViews provides convenient visual ways to enter data series from the keyboard or from disk files, to create new series from existing ones, to display and print series, and to carry out statistical analysis of the relationships among series.

EViews takes advantage of the visual features of modern Windows software. You can use your mouse to guide the operation with standard Windows menus and dialogs. Results appear in windows and can be manipulated with standard Windows techniques.

Alternatively, you may use EViews' powerful command and batch processing language. You can enter and edit commands in the command window. You can create and store the commands in programs that document your research project for later execution.

Installing and Running EViews

Your copy of EViews 5 is distributed on a single CD-ROM. Installation is straightforward—simply insert your CD-ROM disc into a drive, wait briefly while the disc spins-up and the setup program launches, and then simply follow the prompts. If the disc does not spin-up, navigate to the drive using Windows Explorer, then click on the Setup icon.

We have also provided more detailed installation instructions in a separate sheet that you should have received with your EViews package. If you did not receive this sheet, please contact our office, or see our website: www.eviews.com.

Windows Basics

In this section, we provide a brief discussion of some useful techniques, concepts, and conventions that we will use in this manual. We urge those who desire more detail to obtain one of the many good books on Windows.

The Mouse

EViews uses both buttons of the standard Windows mouse. Unless otherwise specified, when we say that you should *click* on an item, we mean a single click of the left mouse-button. *Double click* means to click the left mouse-button twice in rapid succession. We will often refer to *dragging* with the mouse; this means that you should click and hold the left mouse-button down while moving the mouse.

Window Control

As you work, you may find that you wish to change the size of a window or temporarily move a window out of the way. Alternatively, a window may not be large enough to display all of your output, so that you want to move within the window in order to see relevant items. Windows provides you with methods for performing each of these tasks.

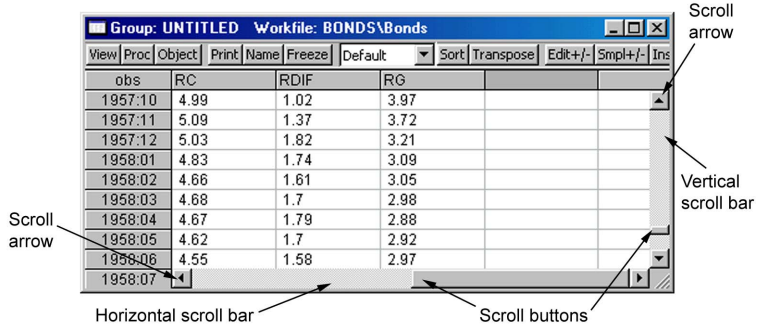
Changing the Active Window

When working in Windows, you may find that you have a number of open windows on your screen. The *active* (top-most) *window* is easily identified since its title bar will generally differ (in color and/or intensity) from the inactive windows. You can make a window active by clicking anywhere in the window, or by clicking on the word **Window** in the main menu, and selecting the window by clicking on its name.

Scrolling

Windows provides both horizontal and vertical *scroll bars* so that you can view information which does not fit inside the window (when all of the information in a window fits inside the viewable area, the scroll bars will be hidden).

The scroll box indicates the overall relative position of the window and the data. Here, the vertical scroll box is near the bottom, indicating that the



window is showing the lower portion of our data. The size of the box also changes to show you the relative sizes of the amount of data in the window and the amount of data that is off-screen. Here, the current display covers roughly half of the horizontal contents of the window.

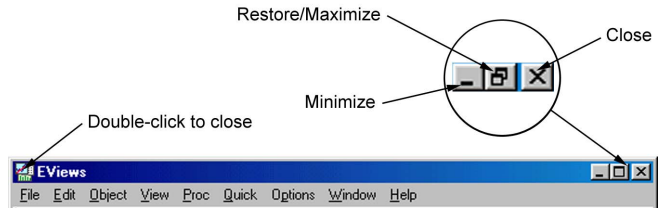
Clicking on the up, down, left, or right scroll arrows on the scroll bar will scroll the display one line in that direction. Clicking on the scroll bar on either side of a scroll box moves the information one screen in that direction.

If you hold down the mouse button while you click on or next to a scroll arrow, you will scroll continuously in the desired direction. To move quickly to any position in the window, drag the scroll box to the desired position.

Minimize/Maximize/Restore/Close

There may be times when you wish to move EViews out of the way while you work in another Windows program. Or you may wish to make the EViews window as large as possible by using the entire display area.

In the upper right-hand corner of each window, you will see a set of buttons which control the window display.



By clicking on the middle (Restore/Maximize) button, you can toggle between using your entire display area for the window, and using the original window size. *Maximize* (🗖) uses your entire monitor display for the application window. *Restore* (🗗)returns the window to its original size, allowing you to view multiple windows. If you are already using the entire display area for your window, the middle button will display the icon for restoring the window, otherwise it will display the icon for using the full screen area.

You can *minimize* your window by clicking on the minimize button in the upper right-hand corner of the window. To *restore* a program that has been minimized, click on the icon in your taskbar.

Lastly, the *close* button provides you with a convenient method for closing the window. To close all of your open EViews windows, you may also select **Window** in the main menu, and either **Close All**, or **Close All Objects**.

Moving and Resizing

You can move or change the size of the window (if it is not maximized or minimized). To move your window, simply click on the title bar (the top of your application window) and drag the window to a new location. To resize, simply put the cursor on one of the four sides or corners of the window. The cursor will change to a double arrow. Drag the window to the desired size, then release the mouse button.

Selecting and Opening Items

To select a single item, you should place the pointer over the item and single click. The item will now be highlighted. If you change your mind, you can change your selection by clicking on a different item, or you can cancel your selection by clicking on an area of the window where there are no items.

You can also select multiple items:

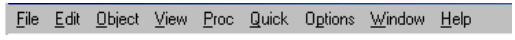
- To select sequential items, click on the first item you want to select, then drag the cursor to the last item you want to select and release the mouse button. All of the items will be selected. Alternatively, you can click on the first item, then hold down the SHIFT key and click on the last item.
- To select non-sequential items, click on the first item you want to select, then while holding the CTRL key, click on each additional item.
- You can also use CTRL-click to "unselect" items which have already been selected. In some cases it may be easier first to select a set of sequential items and then to unselect individual items.

Double clicking on an item will usually open the item. If you have multiple items selected, you can double click anywhere in the highlighted area.

Menus and Dialogs

Windows commands are accessed via *menus*. Most applications contain their own set of menus, which are located on the menu bar along the top of the application window. There are generally drop-down menus associated with the items in the main menu bar.

For example, the main EViews menu contains:



Selecting **File** from this menu will open a drop-down menu containing additional commands. We will describe the EViews menus in greater detail in the coming sections.

There are a few conventions which Windows uses in its menus that are worth remembering:

- A grayed-out command means the command is not currently available.
- An ellipse (...) following the command means that a dialog box (prompting you for additional input) will appear before the command is executed.
- A right-triangle (‣) means that additional (cascading) menus will appear if you select this item.
- A check mark (✔) indicates that the option listed in the menu is currently in effect. If you select the item again, the option will no longer be in effect and the check mark will be removed. This behavior will be referred to as *toggling*.
- Most menu items contain underlined characters representing keyboard shortcuts. You can use the keyboard shortcuts to the commands by pressing the ALT key, and then the underlined character. For example, ALT-F in EViews brings up the **File** drop-down menu.
- If you wish to close a menu without selecting an item, simply click on the menu name, or anywhere outside of the menu. Alternatively, you can press the ESC key.

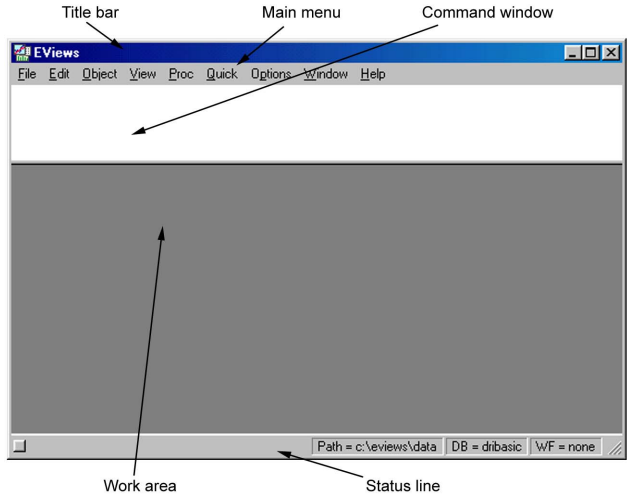
We will often refer to entering information in *dialogs*. Dialogs are boxes that prompt for additional input when you select certain menu items. For example, when you select the menu item to run a regression, EViews opens a dialog prompting you for additional information about the specification, while providing default suggestions for various options. You can always tell when a menu item opens a dialog by the ellipses in the drop-down menu entry.

Break/Cancel

EViews follows the Windows standard in using the ESC key as the break key. If you wish to cancel the current task or ongoing operation, simply press ESC.

The EViews Window

If the program is installed correctly, you should see the EViews window when you launch the program.



You should familiarize yourself with the following main areas in the EViews window.

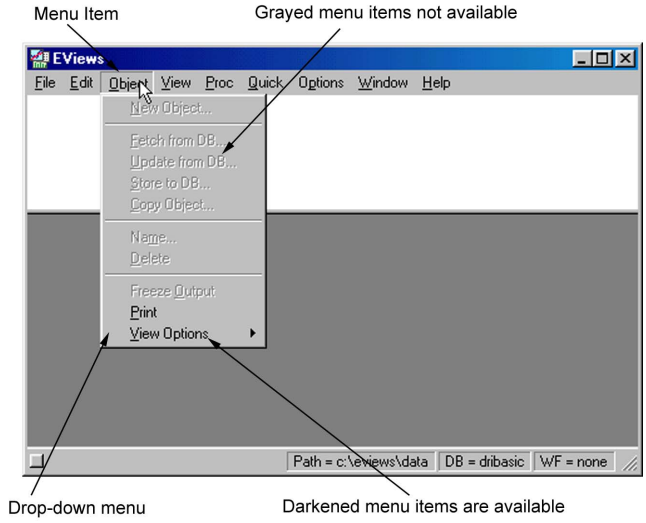
The Title Bar

The *title bar*, labeled **EViews**, is at the very top of the main window. When EViews is the active program in Windows, the title bar has a color and intensity that differs from the other windows (generally it is darker). When another program is active, the EViews title bar will be lighter. If another program is active, EViews may be made active by clicking anywhere in the EViews window or by using ALT-TAB to cycle between applications until the EViews window is active.

The Main Menu

Just below the title bar is the *main menu*. If you move the cursor to an entry in the main menu and click on the left mouse button, a *drop-down menu* will appear. Clicking on an entry in the drop-down menu selects the highlighted item.

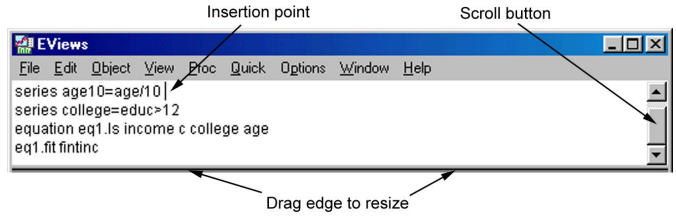
For example, here we click on the **Object** entry in the main menu to reveal a drop-down menu. Notice that some of the items in the drop-down menu are listed in black and others are in gray. In menus, black items may be executed while the gray items are not available. In this example, you cannot create a **New Object** or **Store** an object, but you can **Print** and **View Options**. We will explain this behavior in our discussion of "The Object Window" on page 78.



The Command Window

Below the menu bar is an area called the *command window*. EViews commands may be typed in this window. The command is executed as soon as you hit ENTER.

The vertical bar in the command window is called the *insertion point*. It shows where the letters that you type on the keyboard will be placed. As with standard word processors, if you have typed something in the command area, you can move the insertion point by pointing to the new location and clicking the mouse. If the insertion point is not visible or your keystrokes are not appearing in the window, it probably means that the command window is not active (not receiving keyboard focus); simply click anywhere in the command window to tell EViews that you wish to enter commands.



To toggle between the active window and the command window, press F5.

You may move the insertion point to previously executed commands, edit the existing command, and then press ENTER to execute the edited version of the command.

The command window supports Windows cut-and-paste so that you can easily move text between the command window, other EViews text windows, and other Windows programs. The contents of the command area may also be saved directly into a text file for later use: make certain that the command window is active by clicking anywhere in the window, and then select **File/Save As…** from the main menu.

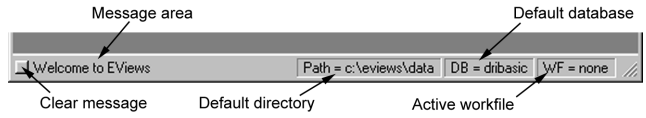
If you have entered more commands than will fit in your command window, EViews turns the window into a standard scrollable window. Simply use the scroll bar or up and down arrows on the right-hand side of the window to see various parts of the list of previously executed commands.

You may find that the default size of the command window is too large or small for your needs. You can resize the command window by placing the cursor at the bottom of the command window, holding down the mouse button and dragging the window up or down. Release the mouse button when the command window is the desired size.

See also "Window and Font Options" on page 937 for a discussion of global settings which affect the use of the command window.

The Status Line

At the very bottom of the window is a *status line* which is divided into several sections.



The left section will sometimes contain status messages sent to you by EViews. These status messages can be cleared manually by clicking on the box at the far left of the status line. The next section shows the *default directory* that EViews will use to look for data and programs. The last two sections display the names of the default database and workfile. In later chapters, we will show you how to change both defaults.

The Work Area

The area in the middle of the window is the work area where EViews will display the various object windows that it creates. Think of these windows as similar to the sheets of paper you might place on your desk as you work. The windows will overlap each other with the foremost window being in *focus* or *active*. Only the active window has a darkened titlebar.

When a window is partly covered, you can bring it to the top by clicking on its titlebar or on a visible portion of the window. You can also cycle through the displayed windows by pressing the F6 or CTRL-TAB keys.

Alternatively, you may select a window by clicking on the **Window** menu item, and selecting the desired name.

You can move a window by clicking on its title bar and dragging the window to a new location. You can change the size of a window by clicking on any corner and dragging the corner to a new location.

Closing EViews

There are a number of ways to close EViews. You can always select **File/Exit** from the main menu, or you can press ALT-F4. Alternatively, you can click on the close box in the upper right-hand corner of the EViews window, or double click on the EViews icon in the upper left-hand corner of the window. If necessary, EViews will warn you and provide you with the opportunity to save any unsaved work.

Where to Go For Help

The EViews Manuals

This *User's Guide* describes how to use EViews to carry out your research. The earlier chapters deal with basic operations, the middle chapters cover basic econometric methods, and the later chapters describe more advanced methods.

Though we have tried to be complete, it is not possible to document every aspect of EViews. There are almost always several ways to do the same thing in EViews, and we cannot describe them all. In fact, one of the strengths of the program is that you will undoubtedly discover alternative, and perhaps more efficient, ways to get your work done.

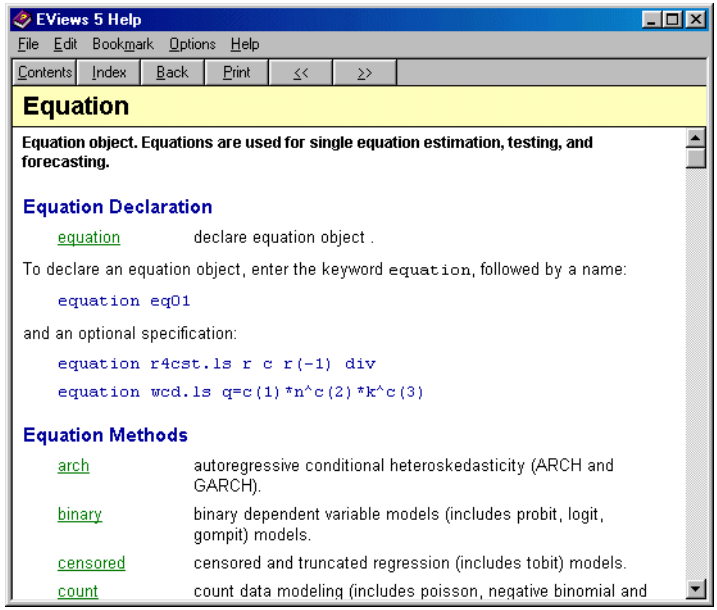
Most of the *User's Guide* explains the visual approach to using EViews. It describes how you can use your mouse to perform operations in EViews. To keep the explanations simple, we do not tell you about alternative ways to get your work done. For example, we will not remind you about the ALT- keyboard alternatives to using the mouse.

When we get to the discussion of the substantive statistical methods available in EViews, we will provide some technical information about the methods, and references to econometrics textbooks and other sources for additional information.

The Help System

Almost all of the EViews documentation may be viewed from within EViews by using the help system. To access the EViews help system, simply go to the main menu and select **Help**.

Since EViews uses standard Windows Help, the on-line manual is fully searchable and hypertext linked. You can set bookmarks to frequently accessed pages, and annotate the on-line documentation with your own notes.


EViews 5 Help
File Edit Bookmark Options Help
Contents Index Back Print << >>
Equation
Equation object. Equations are used for single equation estimation, testing, and forecasting.
Equation Declaration
equation declare equation object .
To declare an equation object, enter the keyword equation, followed by a name:
equation eq01
and an optional specification:
equation r4cst.ls r c r(-1) div
equation wcd.ls q=c(1)*n^c(2)*k^c(3)
Equation Methods
arch autoregressive conditional heteroskedasticity (ARCH and GARCH).
binary binary dependent variable models (includes probit, logit, gompit) models.
censored censored and truncated regression (includes tobit) models.
count count data modeling (includes poisson, negative binomial and

In addition, the Help system will contain updates to the documentation that were made after the manuals went to press.

The World Wide Web

To supplement the information provided in the manuals and the help system, we have set up information areas on the Web that you may access using your favorite browser. You can find answers to common questions about installing, using, and getting the most out of EViews.

Another popular area is our Download Section, which contains on-line updates to EViews 5, sample data and programs, and much more. Your purchase of EViews provides you with much more than the enclosed program and printed documentation. As we make minor changes and revisions to the current version of EViews, we will post them on our web site for you to download. As a valued QMS customer, you are free to download updates to the current version as often as you wish.

So set a bookmark to our site and visit often; the address is:

http://www.eviews.com.

Chapter 2. A Demonstration

In this chapter, we provide a demonstration of some basic features of EViews. The demonstration is meant to be a brief introduction to EViews; not a comprehensive description of the program. A full description of the program begins in Chapter 4, "Object Basics", on page 73.

This demo takes you through the following steps:

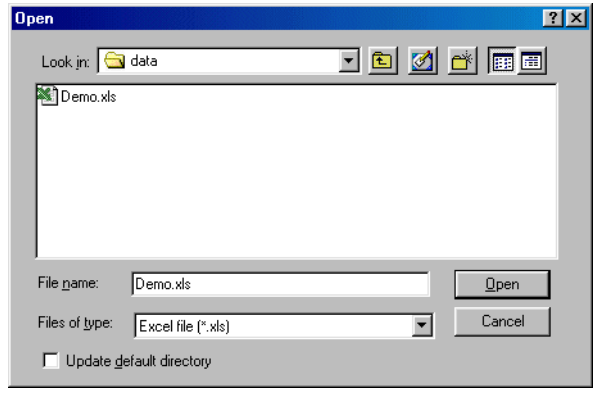
- getting data into EViews from an Excel spreadsheet
- examining your data and performing simple statistical analyses
- using regression analysis to model and forecast a statistical relationship
- performing specification and hypothesis testing
- plotting results

Getting Data into EViews

The first step in most projects will be to read your data into an EViews workfile. EViews provides sophisticated tools for reading from a variety of common data formats, making it extremely easy to get started.

Before we describe the process of reading a foreign data file, note that the data for this demonstration have been included in both Excel spreadsheet and EViews workfile formats in your EViews installation directory ("./Example Files/Data"). If you wish to skip the discussion of opening foreign files, going directly to the analysis part of the demonstration, you may load the EViews workfile by selecting **File/Open/Foreign Data as Workfile...** and opening DEMO.WF1.

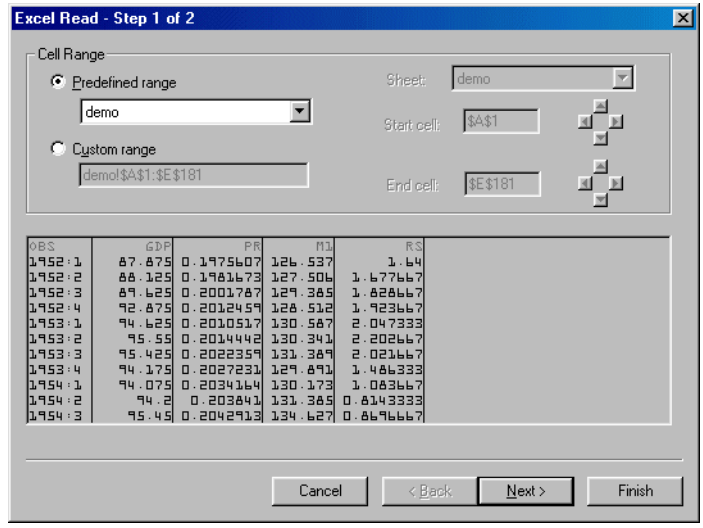
The easiest way to open the Excel file DEMO.XLS, is to drag-and-drop the file into an open EViews application window. You may also drag-and-drop the file onto the EViews icon. Windows will first start the EViews application and will then open the demonstration Excel workfile.



Alternately, you may use the **File/Open/EViews workfile...** dialog, selecting **Files of type** Excel and selecting the desired file.

As EViews opens the file, the program determines that the file is in Excel file format, analyzes the contents of the file, and opens the **Excel Read** wizard.

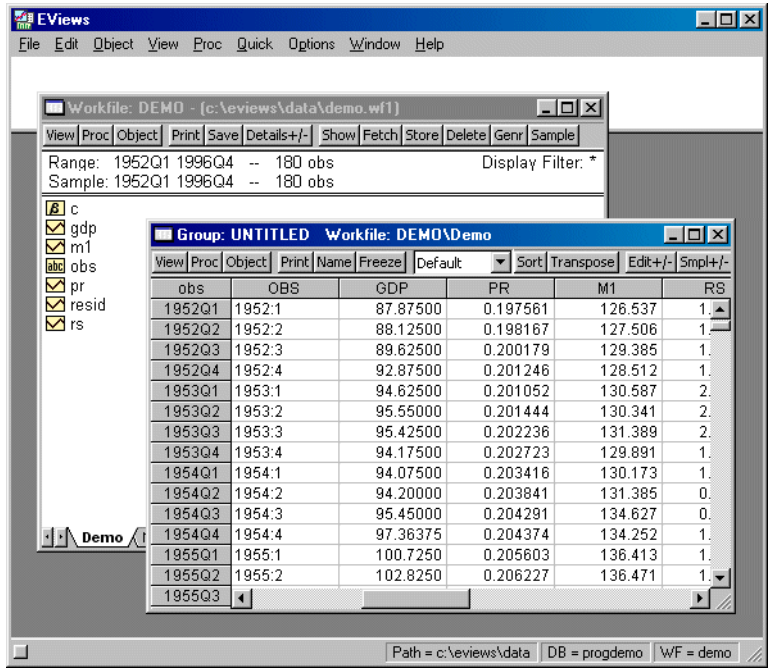
The first page of the wizard includes a preview of the data found in the spreadsheet. In most cases, you need not worry about any of the options on this page. In more complicated cases, you may use the options on this page to provide a custom range of cells to read, or to select a different sheet in the workbook.



The second page of the wizard contains various options for reading the Excel data. These options are set at the most likely choices given the EViews analysis of the contents of your workbook. In most cases, you should simply click on **Finish** to accept the default settings. In other cases where the preview window does not correctly display the desired data, you may click on **Next** and adjust the options that appear on the second page of the wizard. In our example, the data appear to be correct, so we simply click on **Finish** to accept the default settings.

When you accept the settings, EViews automatically creates a workfile that is sized to hold the data, and imports the series into the workfile. The workfile ranges from 1952 quarter 1 to 1996 quarter 4, and contains five series (GDP, M1, OBS, PR, and RS) that you have read from the Excel file. There are also two objects, the coefficient vector C and the series RESID, that are found in all EViews workfiles.

In addition, EViews opens the imported data in a spreadsheet view, allowing you to perform a initial examination of your data. You should compare the spreadsheet views with the Excel worksheet to ensure that the data have been read correctly. You can use the scroll bars and scroll arrows on the right side of the window to view and verify the reminder of the data.



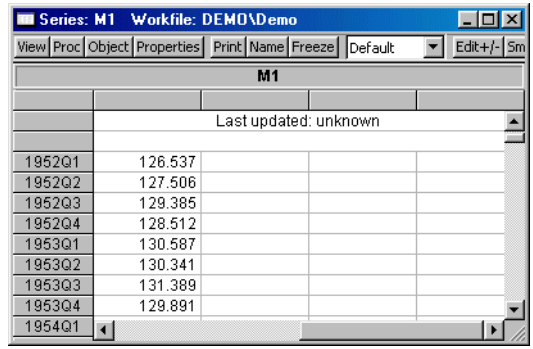
You may wish to click on **Name** in the group toolbar to provide a name for your UNTITLED group. Enter the name ORIGINAL, and click on **OK** to accept the name.

Once you are satisfied that the data are correct, you should save the workfile by clicking on the **Save** button in the workfile window. A saved dialog will open, prompting you for a workfile name and location. You should enter DEMO2.WF1, and then click **OK**. A second dialog may be displayed prompting you to set storage options. Click **OK** to accept the defaults. EViews will save the workfile in the specified directory with the name DEMO2.WF1. A saved workfile may be opened later by selecting **File/Open/Workfile…** from the main menu.

Examining the Data

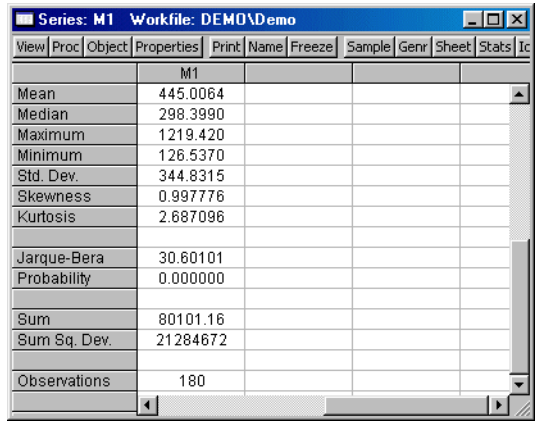
Now that you have your data in an EViews workfile, you may use basic EViews tools to examine the data in your series and groups in a variety of ways.

First, we examine the characteristics of individual series. To see the contents of the M1 series, simply double click on the M1 icon in the workfile window, or select **Quick/Show...** in the main menu, enter `m1`, and click **OK**.



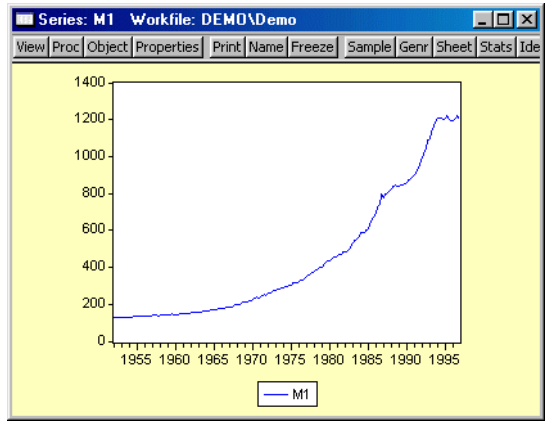
EViews will open the M1 series object and will display the default spreadsheet view of the series. Note the description of the contents of the series ("Series: M1") in the upper leftmost corner of the series window toolbar, indicating that you are working with the M1 series.

You will use the entries in the **View** and **Proc** menus to examine various characteristics of the series. Simply click on the buttons on the toolbar to access these menu entries, or equivalently, select **View** or **Proc** from the main menu.



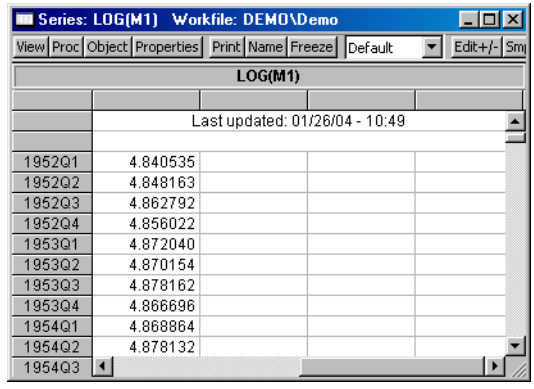
To compute, for example, a table of basic descriptive statistics for M1, simply click on the **View** button, then select **Descriptive Statistics/Stats Table**. EViews will compute descriptive statistics for M1 and change the series view to display a table of results.

Similarly, to examine a line graph of the series, simply select **View/Graph/Line**. EViews will change the M1 series window to display a line graph of the data in the M1 series.



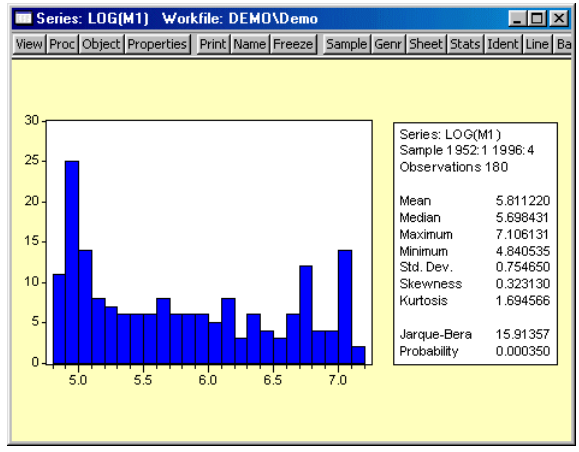
At this point, you may wish to explore the contents of the **View** and **Proc** menus in the M1 series window to see the various tools for examining and working with series data. You may always return to the spreadsheet view of your series by selecting **View/Spreadsheet** from the toolbar or main menu.

Since our ultimate goal is to perform regression analysis with our data expressed in natural logarithms, we may instead wish to work with the log of M1. Fortunately, EViews allows you to work with expressions involving series as easily as you work with the series themselves. To open a series containing this expression, select **Quick/Show…** from the main menu, enter the text for the expression, `log(m1)`, and click **OK**. EViews will open a series window for containing LOG(M1). Note that the titlebar for the series shows that we are working with the desired expression.

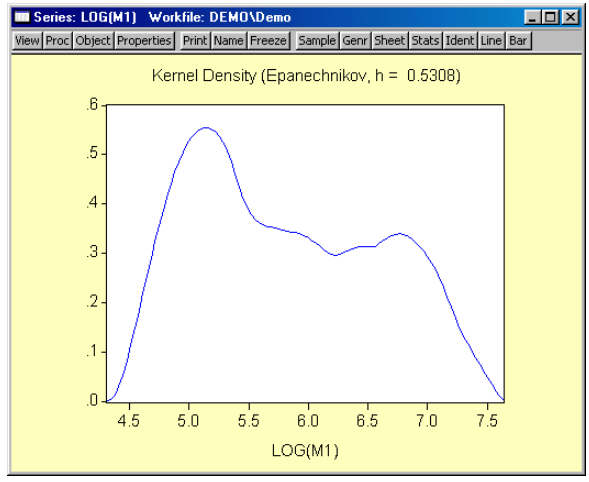


You may work with this auto-series in exactly the same way you worked with M1 above. For example, clicking on **View** in the series toolbar and selecting **Descriptive Statistics/**

Histogram and Stats displays a view containing a histogram and descriptive statistics for LOG(M1):



Alternately, we may display a smoothed version of the histogram by selecting **View/Distribution Graphs/Kernel Density...** and clicking on **OK** to accept the default options:



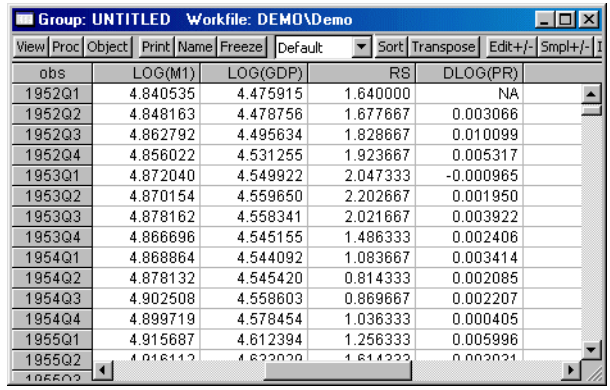
Suppose that you wish to examine multiple series or series expressions. To do so, you will need to construct a group object that contains the series of interest.

Earlier, you worked with an EViews created group object containing all of the series read from your Excel file. Here, we will construct a group object containing expressions involving a subset of those series. We wish to create a group object containing the logarithms of the series M1 and GDP, the level of RS, and the first difference of the logarithm of the

series PR. Simply select **Quick/Show...** from the main EViews menu, and enter the list of expressions and series names:

```
log(m1) log(gdp) rs dlog(pr)
```

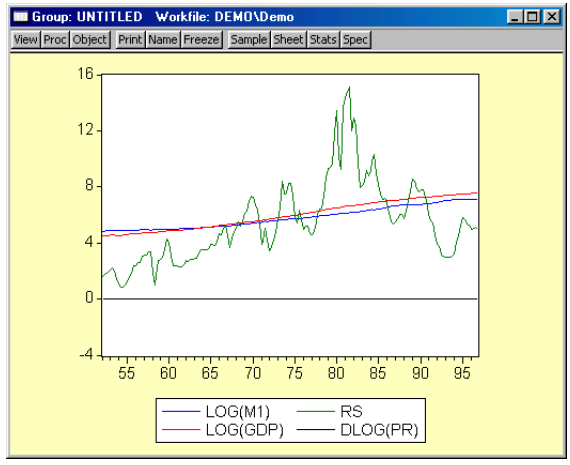
Click on **OK** to accept the input. EViews will open a group window containing a spreadsheet view of the series and expressions of interest.



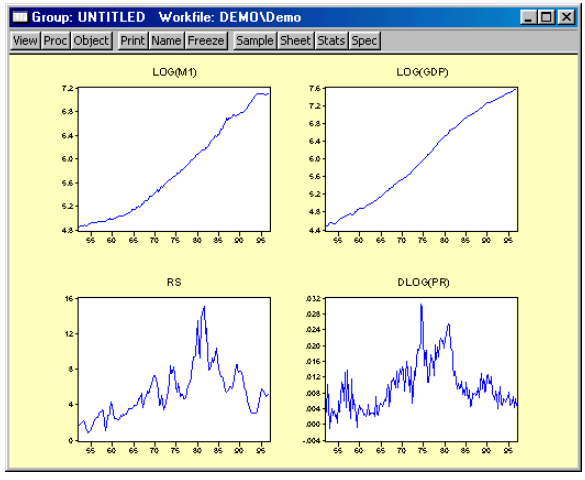
obs	LOG(M1)	LOG(GDP)	RS	DLOG(PR)
1952Q1	4.840535	4.475915	1.640000	NA
1952Q2	4.848163	4.478756	1.677667	0.003066
1952Q3	4.862792	4.495634	1.828667	0.010099
1952Q4	4.856022	4.531255	1.923667	0.005317
1953Q1	4.872040	4.549922	2.047333	-0.000965
1953Q2	4.870154	4.559650	2.202667	0.001950
1953Q3	4.878162	4.558341	2.021667	0.003922
1953Q4	4.866696	4.545155	1.486333	0.002406
1954Q1	4.868864	4.544092	1.083667	0.003414
1954Q2	4.878132	4.545420	0.814333	0.002085
1954Q3	4.902508	4.558603	0.869667	0.002207
1954Q4	4.899719	4.578454	1.036333	0.000405
1955Q1	4.915687	4.612394	1.256333	0.005996
1955Q2	4.916112	4.633029	1.614333	0.003031

As with the series object, you will use the **View** and **Proc** menus of the group to examine various characteristics of the group of series. Simply click on the buttons on the toolbar to access these menu entries or select **View** or **Proc** from the main menu to call up the relevant entries. Note that the entries for a group object will differ from those for a series object since the kinds of operations you may perform with multiple series differ from the types of operations available when working with a single series.

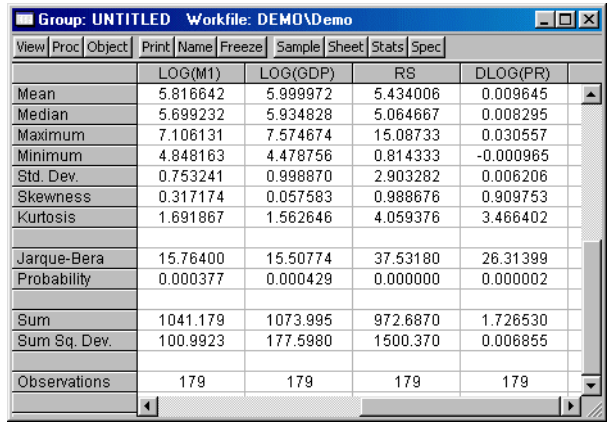
For example, you may select **View/Graphs/Line** from the group object toolbar to display a single graph containing line plots of each of the series in the group:



Alternately, you may select **View/Multiple Graphs/Line** to display the same information, but with each series expression plotted in an individual graph:



Likewise, you may select **View/Descriptive Stats/Individual Samples** to display a table of descriptive statistics computed for each of the series in the group:



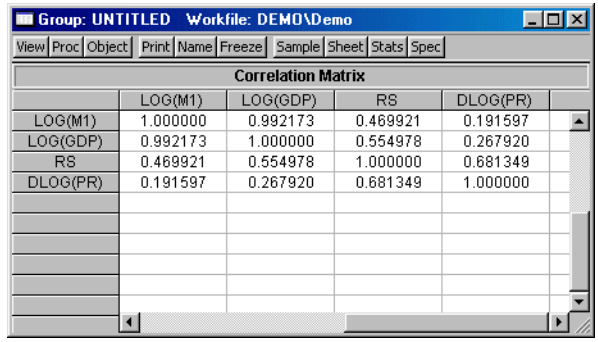
Group: UNTITLED Workfile: DEMO\Demo

View | Proc | Object | Print | Name | Freeze | Sample | Sheet | Stats | Spec

	LOG(M1)	LOG(GDP)	RS	DLOG(PR)
Mean	5.816642	5.999972	5.434006	0.009645
Median	5.699232	5.934828	5.064667	0.008295
Maximum	7.106131	7.574674	15.08733	0.030557
Minimum	4.848163	4.478756	0.814333	-0.000965
Std. Dev.	0.753241	0.998870	2.903282	0.006206
Skewness	0.317174	0.057583	0.988676	0.909753
Kurtosis	1.691867	1.562646	4.059376	3.466402
Jarque-Bera	15.76400	15.50774	37.53180	26.31399
Probability	0.000377	0.000429	0.000000	0.000002
Sum	1041.179	1073.995	972.6870	1.726530
Sum Sq. Dev.	100.9923	177.5980	1500.370	0.006855
Observations	179	179	179	179

Note that the number of observations used for computing descriptive statistics for DLOG(PR) is one less than the number used to compute the statistics for the other expressions. By electing to compute our statistics using "Individual Samples", we informed EViews that we wished to use the series specific samples in each computation, so that the loss of an observation in DLOG(PR) to differencing should not affect the samples used in calculations for the remaining expressions.

We may instead choose to use "Common Samples" so that observations are only used if the data are available for all of the series in the group. Click on **View/Correlations/Common Samples** to display the correlation matrix of the four series for the 179 common observations:



Group: UNTITLED Workfile: DEMO\Demo

View | Proc | Object | Print | Name | Freeze | Sample | Sheet | Stats | Spec

Correlation Matrix

	LOG(M1)	LOG(GDP)	RS	DLOG(PR)
LOG(M1)	1.000000	0.992173	0.469921	0.191597
LOG(GDP)	0.992173	1.000000	0.554978	0.267920
RS	0.469921	0.554978	1.000000	0.681349
DLOG(PR)	0.191597	0.267920	0.681349	1.000000

Once again, we suggest that you may wish to explore the contents of the **View** and **Proc** menus for this group to see the various tools for examining and working with sets of series You can always return to the spreadsheet view of the group by selecting **View/Spreadsheet**.

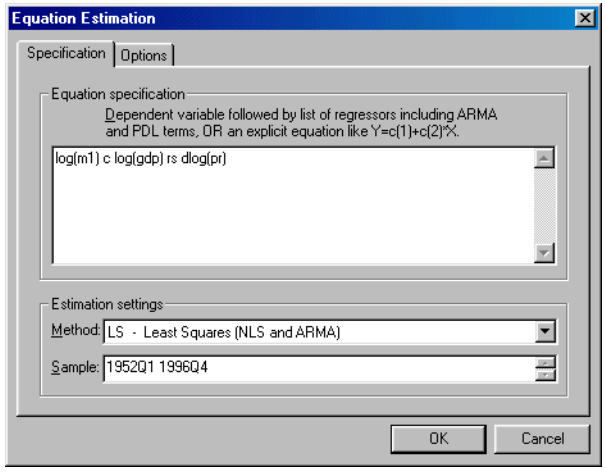
Estimating a Regression Model

We now estimate a regression model for M1 using data over the period from 1952Q1–1992Q4 and use this estimated regression to construct forecasts over the period 1993Q1–2003Q4. The model specification is given by:

$$\log(M1_t) = \beta_1 + \beta_2 \log(\text{GDP}_t) + \beta_3 \text{RS}_t + \beta_4 \Delta\log(\text{PR}_t) + \epsilon_t \tag{2.1}$$

where log(M1) is the logarithm of the money supply, log(GDP) is the log of income, RS is the short term interest rate, and $\Delta\log(\text{PR})$ is the log first difference of the price level (the approximate rate of inflation).

To estimate the model, we will create an equation object. Select **Quick** from the main menu and choose **Estimate Equation…** to open the estimation dialog. Enter the following equation specification:


Equation Estimation
Specification
Options
Equation specification
Dependent variable followed by list of regressors including ARMA and PDL terms, OR an explicit equation like Y=c(1)+c(2)*X.
log(m1) c log(gdp) rs dlog(pr)
Estimation settings
Method: LS - Least Squares (NLS and ARMA)
Sample: 1952Q1 1996Q4
OK
Cancel

Here we list the expression for the dependent variable, followed by the expressions for each of the regressors, separated by spaces. The built-in series name C stands for the constant in the regression.

The dialog is initialized to estimate the equation using the **LS - Least Squares** method for the sample 1952Q1 1996Q4. You should change text in the **Sample** edit box to "1952Q1 1992Q4" to estimate the equation for the subsample of observations.

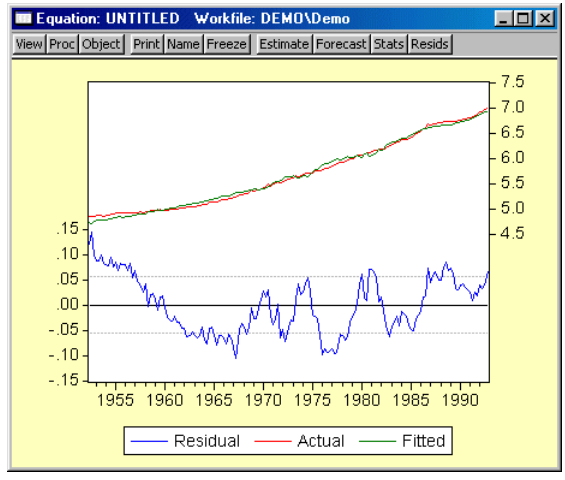
Click **OK** to estimate the equation using least squares and to display the regression results:

Dependent Variable: LOG(M1)
Method: Least Squares
Date: 01/26/04 Time: 13:55
Sample (adjusted): 1952Q2 1992Q4
Included observations: 163 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	1.312383	0.032199	40.75850	0.0000
LOG(GDP)	0.772035	0.006537	118.1092	0.0000
RS	-0.020686	0.002516	-8.221196	0.0000
DLOG(PR)	-2.572204	0.942556	-2.728967	0.0071

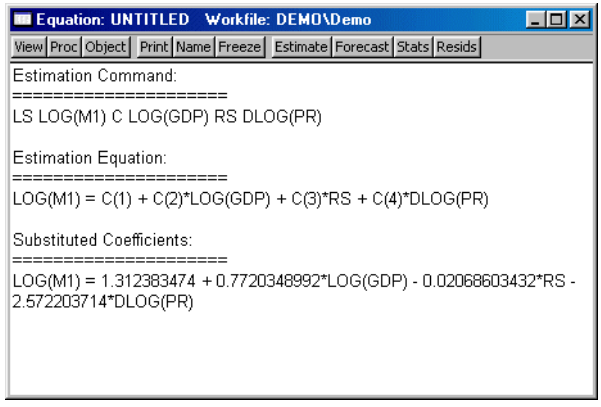
R-squared	0.993274	Mean dependent var	5.692279
Adjusted R-squared	0.993147	S.D. dependent var	0.670253
S.E. of regression	0.055485	Akaike info criterion	-2.921176
Sum squared resid	0.489494	Schwarz criterion	-2.845256
Log likelihood	242.0759	F-statistic	7826.904
Durbin-Watson stat	0.140967	Prob(F-statistic)	0.000000

Note that the equation is estimated from 1952Q2 to 1992Q4 since one observation is dropped from the beginning of the estimation sample to account for the DLOG difference term. The estimated coefficients are statistically significant, with t-statistic values well in excess of 2. The overall regression fit, as measured by the R^2 value, indicates a very tight fit. You can select **View/Actual, Fitted, Residual/Graph** in the equation toolbar to display a graph of the actual and fitted values for the dependent variable, along with the residuals:



Specification and Hypothesis Tests

We can use the estimated equation to perform hypothesis tests on the coefficients of the model. For example, to test the hypothesis that the coefficient on the price term is equal to 2, we will perform a Wald test. First, determine the coefficient of interest by selecting **View/Representations** from the equation toolbar:


Equation: UNTITLED Workfile: DEMO\Demo
View Proc Object Print Name Freeze Estimate Forecast Stats Resids
Estimation Command:
LS LOG(M1) C LOG(GDP) RS DLOG(PR)
Estimation Equation:
LOG(M1) = C(1) + C(2)*LOG(GDP) + C(3)*RS + C(4)*DLOG(PR)
Substituted Coefficients:
LOG(M1) = 1.312383474 + 0.7720348992*LOG(GDP) - 0.02068603432*RS - 2.572203714*DLOG(PR)

Note that the coefficients are assigned in the order that the variables appear in the specification so that the coefficient for the PR term is labeled C(4). To test the restriction on C(4) you should select **View/Coefficient Tests/Wald–Coefficient Restrictions...**, and enter the restriction `c(4)=2`. EViews will report the results of the Wald test:

Wald Test:
Equation: Untitled

Test Statistic	Value	df	Probability
F-statistic	23.53081	(1, 159)	0.0000
Chi-square	23.53081	1	0.0000

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
-2 + C(4)	-4.572204	0.942556

Restrictions are linear in coefficients.

The low probability values indicate that the null hypothesis that C(4) = 2 is strongly rejected.

We should, however, be somewhat cautious of accepting this result without additional analysis. The low value of the Durbin-Watson statistic reported above is indicative of the presence of serial correlation in the residuals of the estimated equation. If uncorrected, serial correlation in the residuals will lead to incorrect estimates of the standard errors, and invalid statistical inference for the coefficients of the equation.

The Durbin-Watson statistic can be difficult to interpret. To perform a more general Breusch-Godfrey test for serial correlation in the residuals, select **View/Residual Tests/ Serial Correlation LM Test…** from the equation toolbar, and specify an order of serial correlation to test against. Entering 1 yields a test against first-order serial correlation:

Breusch-Godfrey Serial Correlation LM Test:

F-statistic	813.0060	Probability	0.000000
Obs*R-squared	136.4770	Probability	0.000000

Test Equation:
Dependent Variable: RESID
Method: Least Squares
Date: 01/26/04 Time: 14:16
Presample missing value lagged residuals set to zero.

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.006355	0.013031	-0.487683	0.6265
LOG(GDP)	0.000997	0.002645	0.376929	0.7067
RS	-0.000567	0.001018	-0.556748	0.5785
DLOG(PR)	0.404143	0.381676	1.058864	0.2913
RESID(-1)	0.920306	0.032276	28.51326	0.0000

R-squared	0.837282	Mean dependent var	5.13E-16
Adjusted R-squared	0.833163	S.D. dependent var	0.054969
S.E. of regression	0.022452	Akaike info criterion	-4.724644
Sum squared resid	0.079649	Schwarz criterion	-4.629744
Log likelihood	390.0585	F-statistic	203.2515
Durbin-Watson stat	1.770965	Prob(F-statistic)	0.000000

The top part of the output presents the test statistics and associated probability values. The test regression used to carry out the test is reported below the statistics.

The statistic labeled “Obs*R-squared” is the LM test statistic for the null hypothesis of no serial correlation. The (effectively) zero probability value strongly indicates the presence of serial correlation in the residuals.

Modifying the Equation

The test results suggest that we need to modify our original specification to take account of the serial correlation.

One approach is to include lags of the independent variables. To add variables to the existing equation, click on the **Estimate** button in the equation toolbar and edit the specification to include lags for each of the original explanatory variables:

```
log(m1) c log(gdp) rs dlog(pr) log(m1(-1)) log(gdp(-1)) rs(-1)
    dlog(pr(-1))
```

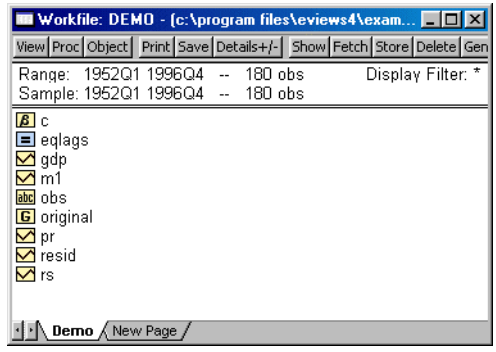
Note that lags are specified by including a negative number, enclosed in parentheses, following the series name. Click on **OK** to estimate the new specification and to display the results:

Dependent Variable: LOG(M1)
Method: Least Squares
Date: 01/26/04 Time: 14:17
Sample (adjusted): 1952Q3 1992Q4
Included observations: 162 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.071297	0.028248	2.523949	0.0126
LOG(GDP)	0.320338	0.118186	2.710453	0.0075
RS	-0.005222	0.001469	-3.554801	0.0005
DLOG(PR)	0.038615	0.341619	0.113036	0.9101
LOG(M1(-1))	0.926640	0.020319	45.60375	0.0000
LOG(GDP(-1))	-0.257364	0.123264	-2.087910	0.0385
RS(-1)	0.002604	0.001574	1.654429	0.1001
DLOG(PR(-1))	-0.071650	0.347403	-0.206246	0.8369

R-squared	0.999604	Mean dependent var	5.697490
Adjusted R-squared	0.999586	S.D. dependent var	0.669011
S.E. of regression	0.013611	Akaike info criterion	-5.707729
Sum squared resid	0.028531	Schwarz criterion	-5.555255
Log likelihood	470.3261	F-statistic	55543.30
Durbin-Watson stat	2.393764	Prob(F-statistic)	0.000000

Note that EViews has automatically adjusted the estimation sample to accommodate the additional lagged variables. We will save this equation in the workfile for later use. Press the **Name** button in the toolbar and name the equation EQLAGS.



The EQLAGS equation object will be placed in the workfile.

One common method of accounting for serial correlation is to include autoregressive (AR) and/or moving average (MA) terms in the equation. To estimate the model with an AR(1) error specification, you should make a copy of the EQLAGS equation by clicking **Object/Copy Object...** in the EQLAGS window. EViews will create a new untitled equation containing all of the information from the previous equation. Press **Estimate** on the toolbar of the copy and modify the specification to read

```
log(m1) c log(gdp) rs dlog(pr) ar(1)
```

This specification removes the lagged terms, replacing them with an AR(1) specification:

$$\begin{aligned} \log(M1_t) &= \beta_1 + \beta_2 \log(\text{GDP}_t) + \beta_3 \text{RS}_t + \beta_4 \Delta\log(\text{PR}_t) + u_t \\ u_t &= \rho u_{t-1} + \epsilon_t \end{aligned} \tag{2.2}$$

Click **OK** to accept the new specification. EViews will estimate the equation and will report the estimation results, including the estimated first-order autoregressive coefficient of the error term:

Dependent Variable: LOG(M1)
Method: Least Squares
Date: 01/26/04 Time: 17:21
Sample (adjusted): 1952Q3 1992Q4
Included observations: 162 after adjusting endpoints
Convergence achieved after 17 iterations

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	1.050283	0.328313	3.199031	0.0017
LOG(GDP)	0.794937	0.049332	16.11418	0.0000
RS	-0.007395	0.001457	-5.075131	0.0000
DLOG(PR)	-0.008018	0.348689	-0.022996	0.9817
AR(1)	0.968109	0.018189	53.22351	0.0000

R-squared	0.999526	Mean dependent var	5.697490
Adjusted R-squared	0.999514	S.D. dependent var	0.669011
S.E. of regression	0.014751	Akaike info criterion	-5.564584
Sum squared resid	0.034164	Schwarz criterion	-5.469288
Log likelihood	455.7313	F-statistic	82748.93
Durbin-Watson stat	2.164286	Prob(F-statistic)	0.000000

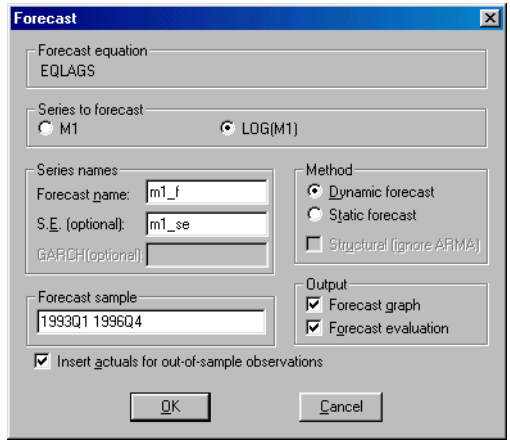
Inverted AR Roots .97

The fit of the AR(1) model is roughly comparable to the lag model, but its somewhat higher values for both the Akaike and the Schwarz information criteria indicate that the previous lag model may be preferred. Accordingly, we will work with the lag model in EQLAGS for the remainder of the demonstration.

Forecasting from an Estimated Equation

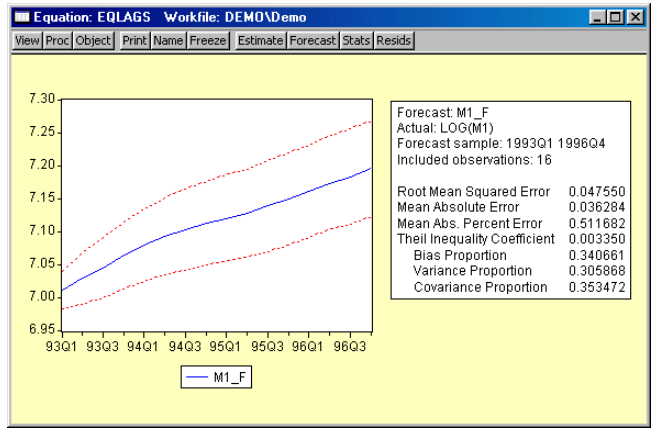
We have been working with a subset of our data, so that we may compare forecasts based upon this model with the actual data for the post-estimation sample 1993Q1–1996Q4.

Click on the **Forecast** button in the EQLAGS equation toolbar to open the forecast dialog:



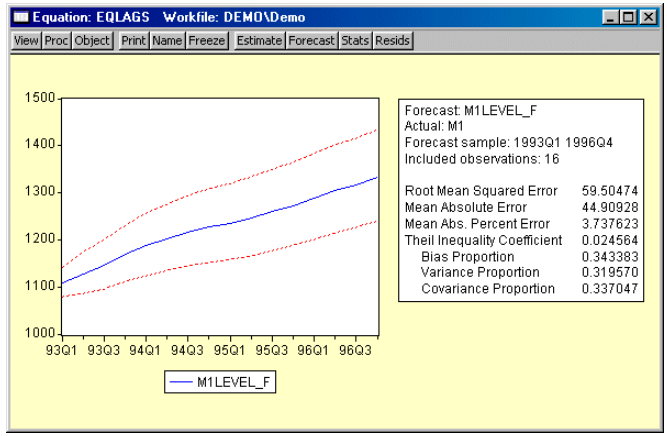
We set the forecast sample to 1993Q1–1996Q4 and provide names for both the forecasts and forecast standard errors so both will be saved as series in the workfile. The forecasted values will be saved in M1_F and the forecast standard errors will be saved in M1_SE.

Note also that we have elected to forecast the log of M1, not the level, and that we request both graphical and forecast evaluation output. The **Dynamic** option constructs the forecast for the sample period using only information available at the beginning of 1993Q1. When you click **OK**, EViews displays both a graph of the forecasts, and statistics evaluating the quality of the fit to the actual data:



Alternately, we may also choose to examine forecasts of the level of M1. Click on the **Forecast** button in the EQLAGS toolbar to open the forecast dialog, and select **M1** under the **Series to forecast** option. Enter a new name to hold the forecasts, say M1LEVEL_F, and

click **OK**. EViews will present a graph of the forecast of the level of M1, along with the asymmetric confidence intervals for this forecast:

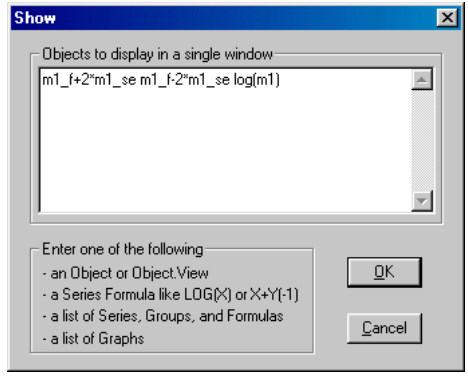


The series that the forecast procedure generates are ordinary EViews series that you may work with in the usual ways. For example, we may use the forecasted series for LOG(M1) and the standard errors of the forecast to plot actuals against forecasted values with (approximate) 95% confidence intervals for the forecasts.

We will first create a new group object containing these values. Select **Quick/Show...** from the main menu, and enter the expressions:

```
m1_f+2*m1_se m1_f-2*m1_se log(m1)
```

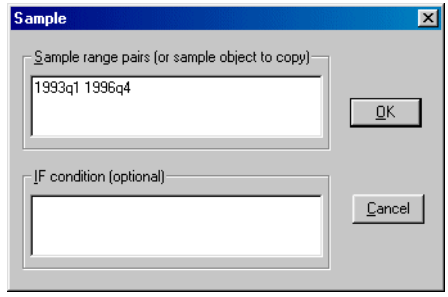
to create a group containing the confidence intervals for the forecast of LOG(M1) and the actual values of LOG(M1):


Show
Objects to display in a single window
m1_f+2*m1_se m1_f-2*m1_se log(m1)
Enter one of the following
- an Object or Object.View
- a Series Formula like LOG(X) or X+Y(-1)
- a list of Series, Groups, and Formulas
- a list of Graphs
OK
Cancel

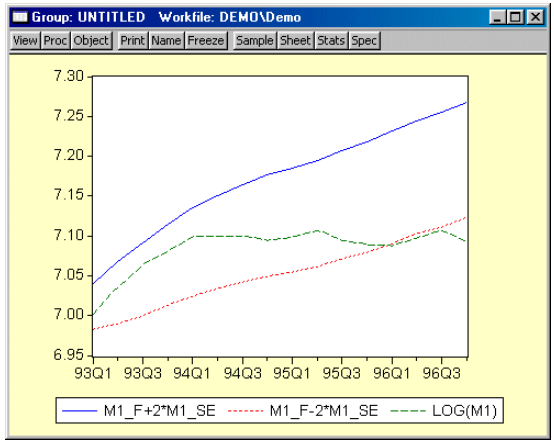
There are three expressions in the dialog. The first two represent the upper and lower bounds of the (approximate) 95% forecast interval as computed by evaluating the values

of the point forecasts plus and minus two times the standard errors. The last expression represents the actual values of the dependent variable.

When you click **OK**, EViews opens an untitled group window containing a spreadsheet view of the data. Before plotting the data, we will change the sample of observations so that we only plot data for the forecast sample. Select **Quick/Sample…** or click on the **Sample** button in the group toolbar, and change the sample to include only the forecast period:



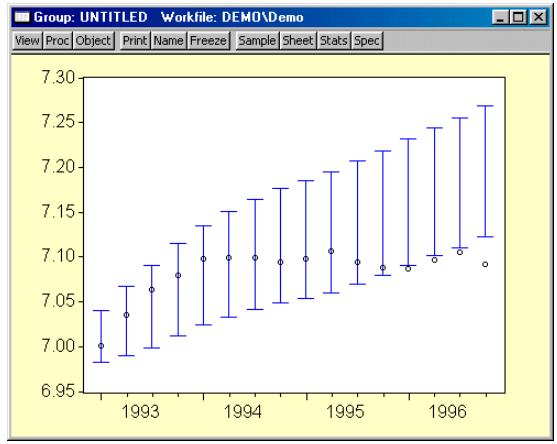
To plot the data for the forecast period, select **View/Graph/Line** from the group window:



The actual values of log(M1) are within the forecast interval for most of the forecast period, but fall below the lower bound of the 95% confidence interval beginning in 1996:1.

For an alternate view of these data, you can select **View/Graph/Error Bar**, which displays the graph as follows:

This graph shows clearly that the forecasts of LOG(M1) over-predict the actual values in the last four quarters of the forecast period.



Additional Testing

Note that the above specification has been selected for illustration purposes only. Indeed, performing various specification tests on EQLAGS suggests that there may be a number of problems with the existing specification.

For one, there is quite a bit of serial correlation remaining even after estimating the lag specification. A test of serial correlation in the EQLAGS equation (by selecting **View/Residual Tests/Serial Correlation LM Test...**, and entering 1 for the number of lags) rejects the null hypothesis of no serial correlation in the reformulated equation:

Breusch-Godfrey Serial Correlation LM Test:			
F-statistic	7.880369	Probability	0.005648
Obs*R-squared	7.935212	Probability	0.004848

Moreover, there is strong evidence of autoregressive conditional heteroskedasticity (ARCH) in the residuals. Select **View/Residual Tests/ARCH LM Test...** and accept the default of 1. The ARCH test results strongly suggest the presence of ARCH in the residuals:

ARCH Test:			
F-statistic	11.21965	Probability	0.001011
Obs*R-squared	10.61196	Probability	0.001124

In addition to serial correlation and ARCH, there is an even more fundamental problem with the above specification since, as the graphs attest, LOG(M1) exhibits a pronounced upward trend, suggesting that we should perform a unit root in this series. The presence of a unit root will indicate the need for further analysis.

We once again display the LOG(M1) series window by clicking on **Window** and selecting the LOG(M1) series window from the menu. If the series window for LOG(M1) is not present (if you previously closed the window), you may again open a new window by selecting **Quick/Show...**, entering `log(m1)`, and clicking **OK**.

Before computing the test statistic, we will reset the workfile sample to all of the observations by clicking on Quick/Sample... and entering `@all` in the dialog.

Next, to perform an Augmented Dickey-Fuller (ADF) test for nonstationarity of this series, select **View/Unit Root Test...** and click on **OK** to accept the default options. EViews will perform an ADF test and display the test results. The top portion of the output reads:

Null Hypothesis: LOG(M1) has a unit root
Exogenous: Constant
Lag Length: 0 (Automatic based on SIC, MAXLAG=3)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-3.684205	0.0157
Test critical values:	1% level	-3.920350	
	5% level	-3.065585	
	10% level	-2.673459	

*MacKinnon (1996) one-sided p-values.
Warning: Probabilities and critical values calculated for 20 observations
and may not be accurate for a sample size of 16

EViews performs the ADF test statistic with the number of lagged difference terms in the test equation (here, zero) determined by automatic selection. The ADF test statistic value has a probability value of 0.0157, providing evidence that we may reject the null hypothesis of a unit root.

If a unit root were present in our data, we may wish to adopt more sophisticated statistical models. These techniques are discussed in Chapter 17, "Time Series Regression" and Chapter 24, "Vector Autoregression and Error Correction Models" which deal with basic time series and vector autoregression and vector error correction specifications, respectively.

Chapter 3. Workfile Basics

Managing the variety of tasks associated with your work can be a complex and time-consuming process. Fortunately, EViews' innovative design takes much of the effort out of organizing your work, allowing you to concentrate on the substance of your project. EViews provides sophisticated features that allow you to work with various types of data in an intuitive and convenient fashion.

Before describing these features, we begin by outlining the basic concepts underlying the EViews approach to working with datasets using workfiles, and describing simple methods to get you started on creating and working with workfiles in EViews.

What is a Workfile?

At a basic level, a *workfile* is simply a container for EViews objects (see Chapter 4, "Object Basics", on page 73). Most of your work in EViews will involve objects that are contained in a workfile, so your first step in any project will be to create a new workfile or to load an existing workfile into memory.

Every workfile contains one or more *workfile pages*, each with its own objects. A workfile page may be thought of as a subworkfile or subdirectory that allows you to organize the data within the workfile.

For most purposes, you may treat a workfile page as though it were a workfile (just as a subdirectory is also a directory) since there is often no practical distinction between the two. Indeed, in the most common setting where a workfile contains only a single page, the two are completely synonymous. Where there is no possibility of confusion, we will use the terms "workfile" and "workfile page" interchangeably.

Workfiles and Datasets

While workfiles and workfile pages are designed to hold a variety of EViews objects, such as equations, graphs, and matrices, their primary purpose is to hold the contents of *datasets*. A dataset is defined here as a data rectangle, consisting of a set of *observations* on one or more *variables*—for example, a time series of observations on the variables GDP, investment, and interest rates, or perhaps a random sample of observations containing individual incomes and tax liabilities.

Key to the notion of a dataset is the idea that each observation in the dataset has a unique *identifier*, or *ID*. Identifiers usually contain important information about the observation, such as a date, a name, or perhaps an identifying code. For example, annual time series data typically use year identifiers ("1990", "1991", ...), while

cross-sectional state data generally use state names or abbreviations ("AL", "AK", ..., "WY"). More complicated identifiers are associated with longitudinal data, where one typically uses *both* an individual ID and a date ID to identify each observation.

Observation IDs are often, but not always, included as a part of the dataset. Annual datasets, for example, usually include a variable containing the year associated with each observation. Similarly, large cross-sectional survey data typically include an interview number used to identify individuals.

In other cases, observation IDs are not provided in the dataset, but external information is available. You may know, for example, that the 21 otherwise unidentified observations in a dataset are for consecutive years beginning in 1990 and continuing to 2010.

In the rare case were there is no additional identifying information, one may simply use a set of default integer identifiers that enumerate the observations in the dataset ("1", "2", "3", ...).

Since the primary purpose of every workfile page is to hold the contents of a single dataset, each page must contain information about observation identifiers. Once identifier information is provided, the workfile page provides context for working with observations in the associated dataset, allowing you to use dates, handle lags, or work with longitudinal data structures.

Creating a Workfile

There are several ways to create and set up a new workfile. The first task you will face in setting up a workfile (or workfile page) is to specify the structure of your workfile. We focus here on three distinct approaches:

First, you may simply *describe the structure of your workfile*. EViews will create a new workfile for you to enter or import your data.

Describing the workfile is the simplest method, requiring only that you answer a few simple questions—it works best when the identifiers follow a simple pattern that is easily described (for example, "annual data from 1950 to 2000" or "quarterly data from 1970Q1 to 2002Q4"). This approach must be employed if you plan to enter data into EViews by typing or copy-and-pasting data.

In the second approach, you simply *open and read data from a foreign data source*. EViews will analyze the data source, create a workfile, and then automatically import your data.

The final approach, which should be reserved for more complex settings, involves two distinct steps. In the first, you create a new workfile using one of the first two approaches (by describing the structure of the workfile, or by opening and reading from a foreign data

source). Next, you will s*tructure the workfile*, by showing EViews how to construct unique identifiers, in some cases by using values of the variables contained in the dataset.

We begin by describing the first two methods. The third approach, involving the more complicated task of structuring a workfile, will be taken up in "Structuring a Workfile" on page 207.

Creating a Workfile by Describing its Structure

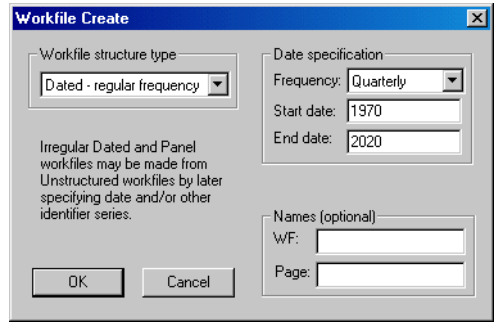
To describe the structure of your workfile, you will need to provide EViews with external information about your observations and their associated identifiers. As examples, you might tell EViews that your dataset consists of a time series of observations for each quarter from 1990Q1 to 2003Q4, or that you have information for every day from the beginning of 1997 to the end of 2001, or that you have a dataset with 500 observations and no additional identifier information.

To create a new workfile, select **File/New Workfile...** from the main menu to open the **Workfile Create** dialog.

On the left side of the dialog is a combo box for describing the underlying structure of your dataset. You will choose between the **Dated - regular frequency**, the **Unstructured**, and the **Balanced Panel** settings. Generally speaking, you should use **Dated - regular frequency** if you have a simple time series dataset, for a simple panel dataset you should use **Balanced Panel**, and in all other cases, you should select **Unstructured**. Additional detail to aid you in making a selection is provided in the description of each category.

Describing a Dated Regular Frequency Workfile

When you select **Dated - regular frequency**, EViews will prompt you to select a frequency for your data. You may choose between the standard EViews supported date frequencies (**Annual**, **Semi-annual**, **Quarterly**, **Monthly**, **Weekly**, **Daily - 5 day week**, **Daily - 7 day week**), and a special frequency (**Integer date**) which is a generalization of a simple enumeration.



In selecting a frequency, you set intervals between observations in your data (whether they are annual, semi-annual, quarterly, monthly, weekly, 5-day daily, or 7-day daily), which allows EViews to use all available calendar information to organize and manage your data. For example, when moving between daily and weekly or annual data, EViews knows that some years contain days in each of 53 weeks, and that some years have 366 days, and will use this information when working with your data.

As the name suggests, *regular frequency* data arrive at regular intervals, defined by the specified frequency (*e.g.*, monthly). In contrast, *irregular frequency* data do not arrive in regular intervals. An important example of irregular data is found in stock and bond prices where the presence of holidays and other market closures ensures that data are observed only irregularly, and not in a regular 5-day daily frequency. Standard macroeconomic data such as quarterly GDP or monthly housing starts are examples of regular data.

EViews also prompts you to enter a **Start date** and **End date** for your workfile. When you click on **OK**, EViews will create a regular frequency workfile with the specified number of observations and the associated identifiers.

Suppose, for example, that you wish to create a quarterly workfile that begins with the first quarter of 1970 and ends in the last quarter of 2020.

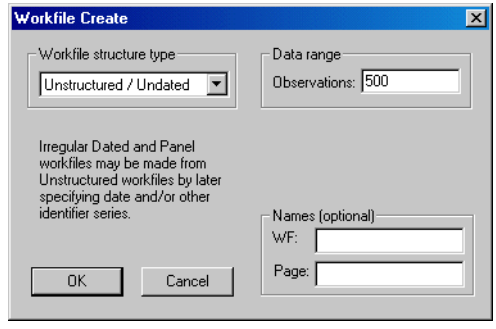
- First, select **Dated - regular frequency** for the workfile structure, and then choose the **Quarterly** frequency.
- Next, enter the **Start date** and **End date**. There are a number of ways to fill in the dates. EViews will use the largest set of observations consistent with those dates, so if you enter "1970" and "2020", your quarterly workfile will begin in the first quarter of 1970, and end in the last quarter of 2020. Entering the date pair "Mar 1970" and "Nov 2020", or the start-end pair "3/2/1970" and "11/15/2020" would have generated a workfile with the same structure, since the implicit start and end quarters are the same in all three cases.

This latter example illustrates a fundamental principle regarding the use of date information in EViews. Once you specify a date frequency for a workfile, EViews will use all available calendar information when interpreting date information. For example, given a quarterly frequency workfile, EViews knows that the date "3/2/1990" is in the first quarter of 1990 (see "Dates" on page 129 of the *Command and Programming Reference* for details).

Lastly, you may optionally provide a name to be given to your workfile and a name to be given to the workfile page.

Describing an Unstructured Workfile

Unstructured data are simply undated data which use the default integer identifiers. You should choose the **Unstructured** type if you wish to create a workfile that uses the default identifiers, or if your data are not a **Dated - regular frequency** or **Balanced Panel**.



When you select this structure in the combo box, the remainder of the dialog will change, displaying a single field prompting you for the number of observations. Enter the number of observations, and click on **OK** to proceed. In the example depicted here, EViews will create a 500 observation workfile containing integer identifiers ranging from 1 to 500.

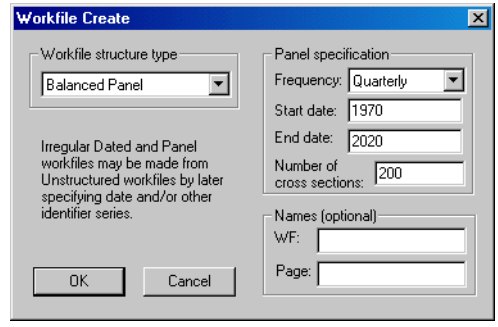
In many cases, the integer identifiers will be sufficient for you to work with your data. In more complicated settings, you may wish to further refine your identifiers. We describe this process in "Applying a Structure to a Workfile" on page 218.

Describing a Balanced Panel Workfile

The **Balanced Panel** entry provides a simple method of describing a regular frequency *panel data* structure. Panel data is the term that we use to refer to data containing observations with both a group (cross-section) and cell (within-group) identifiers.

This entry may be used when you wish to create a balanced structure in which every cross-section follows the same regular frequency with the same date observations. Only the barest outlines of the procedure are provided here since a proper discussion requires a full description of panel data and the creation of the advanced workfile structures. Panel data and structured workfiles are discussed at length in "Structuring a Workfile" on page 207.

To create a balanced panel, select **Balanced Panel** in the combo box, specify the desired **Frequency**, and enter the **Start date** and **End date**, and **Number of cross sections**. You may optionally name the workfile and the workfile page. Click on **OK**. EViews will create a balanced panel workfile of the given frequency, using the specified start and end dates and number of cross-sections.



Here, EViews creates a 200 cross-section, regular frequency, quarterly panel workfile with observations beginning in 1970Q1 and ending in 2020Q4.

Unbalanced panel workfiles or workfiles involving more complex panel structures should be created by first defining an unstructured workfile, and then applying a panel workfile structure.

Creating a Workfile by Reading from a Foreign Data Source

A second method of creating an EViews workfile is to open a foreign (non-EViews format) data source and to read the data into an new EViews workfile.

The easiest way to read foreign data into a new workfile is to copy the foreign data source to the Windows clipboard, right click on the gray area in your EViews window, and select **Paste as new Workfile**. EViews will automatically create a new workfile containing the contents of the clipboard. Such an approach, while convenient, is only practical for small amounts of data.

Alternately, you may open a foreign data source as an EViews workfile. To open a foreign data source, first select **File/Open/Foreign Data as Workfile...**, to bring up the standard file **Open** dialog. Clicking on the **Files of type** combo box brings up a list of the file types that EViews currently supports for opening a workfile.

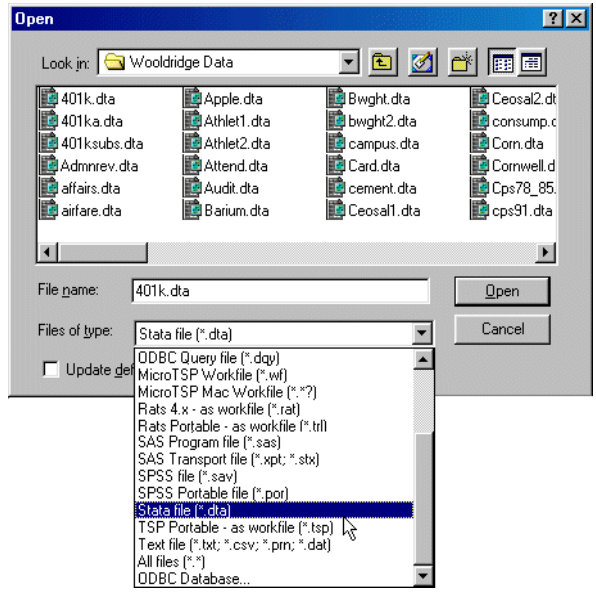
If you select a time series database file (Aremos TSD, GiveWin/Pc-Give, Rats 4.x, Rats Portable, TSP Portable), EViews will create a new, regular frequency workfile containing the contents of the entire file. If there are mixed frequencies in the database, EViews will select the lowest frequency, and convert all of the series to that frequency using the default conversion settings (we emphasize here that all of these database formats may also be opened as databases by selecting **File/Open/Database...** and filling out the dialogs, allowing for additional control over the series to be read, the new workfile frequency, and any frequency conversion).

If you choose one of the remaining source types, EViews will create a new unstructured workfile. First, EViews will open a series of dialogs prompting you to describe and select data to be read. The data will be read into the new workfile, which will be resized to fit. If there is a single date series in the data, EViews will attempt to restructure the workfile using the date series. If this is not possible but you still wish to use dates with these data, you will have to define a structured workfile using the advanced workfile structure tools (see "Structuring a Workfile" beginning on page 207).

The import as workfile interface is available for Microsoft Access files, Gauss Dataset files, ODBC Dsn files, ODBC Query files, SAS Transport files, native SPSS files (using the SPSS Input/output .DLL that should be installed on your system), SPSS Portable files, Stata files, Excel files, raw ASCII or binary files, or ODBC Databases and queries (using the ODBC driver already present on your system).

An Illustration

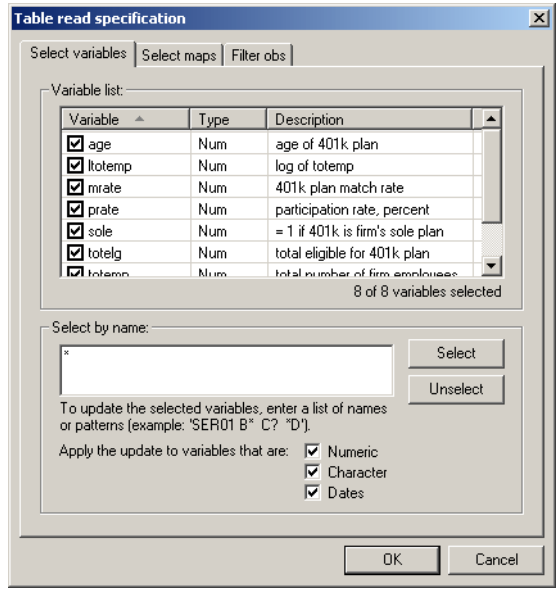
We will use a Stata file to illustrate the basic process of creating a new workfile (or a workfile page) by opening a foreign source file.



To open the file, first navigate to the appropriate directory and select **Stata file** to display available files of that type. Next, double-click on the name to select and open the file, or enter the filename in the dialog and click on **Open** to accept the selection.

A simple alternative to opening the file from the menu is to drag-and-drop your foreign file into the EViews window.

EViews will open the selected file, validate its type, and will display a tabbed dialog allowing you to select the specific data that you wish to read into your new workfile. If you wish to read all of your data using the default settings, click on **OK** to proceed. Otherwise you may use each of the tabs to change the read behavior.



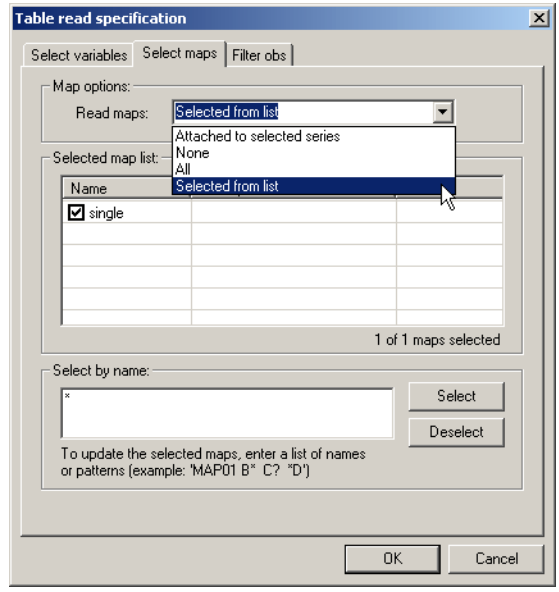
The **Select variables** tab of the dialog should be used to choose the series data to be included. The upper list box shows the names of the variables that can be read into EViews series, along with the variable data type, and if available, a description of the data. The variables are first listed in the order in which they appear in the file. You may choose to sort the data by clicking on the header for the column. The display will be toggled between three states: the original order, sorted (ascending), and sorted (descending). In the latter two cases, EViews will display a

small arrow on the header column indicating the sort type. Here, the data are sorted by variable name in ascending order.

When the dialog first opens, all variables are selected for reading. You can change the current state of any variable by checking or unchecking the corresponding checkbox. The number of variables selected is displayed at the bottom right of the list.

There may be times when checking and unchecking individual variables is inconvenient (*e.g.*, when there are thousands of variable names). The bottom portion of the dialog provides you with a control that allows you to select or unselect variables by name. Simply enter the names of variables using wildcard characters if desired, choose the types of interest, and click on the appropriate button. For example, entering "A* B?" in the selection edit box, selecting only the **Numeric** checkbox, and clicking on **Unselect** will uncheck all numeric series beginning with the letter "A" and all numeric series with two character names beginning in "B".

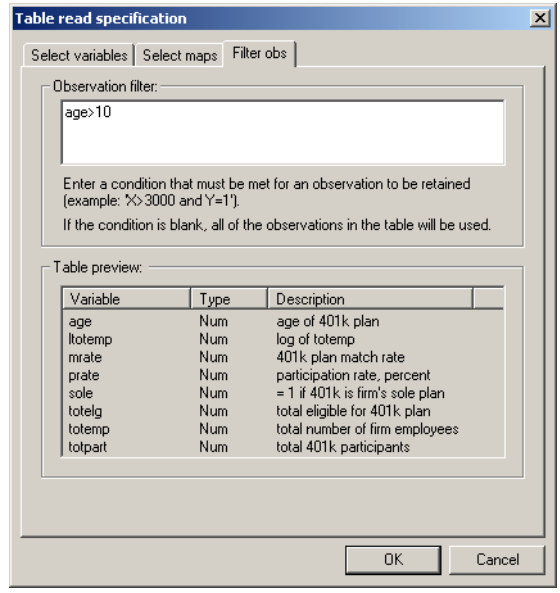
When opening datasets that contain value labels, EViews will display a second tabbed dialog page labeled **Select maps**, which controls the importing of value maps. On this page, you will specify how you wish EViews to handle these value labels. You should bear in mind that when opening datasets which do not contain value labels, EViews will not display the value map tab.



The upper portion of the dialog contains a combo box where you specify which labels to read. You may choose between the default **Attached to selected series**, **None**, **All**, or **Selected from list**.

The selections should be self-explanatory—**Attached to selected series** will only load maps that are used by the series that you have selected for inclusion; **Selected from list** (depicted) displays a map selection list in which you may check and uncheck individual label names along with a control to facilitate selecting and deselecting labels by name.

Lastly, the **Filter obs** page brings up an observation filter specification where you may enter a condition on your data that must be met for a given observation to be read.



When reading the dataset, EViews will discard any observation that does not meet the specified criteria. Here we tell EViews that we only wish to keep observations where AGE > 10.

Once you have specified the characteristics of your table read, click on **OK** to begin the procedure.

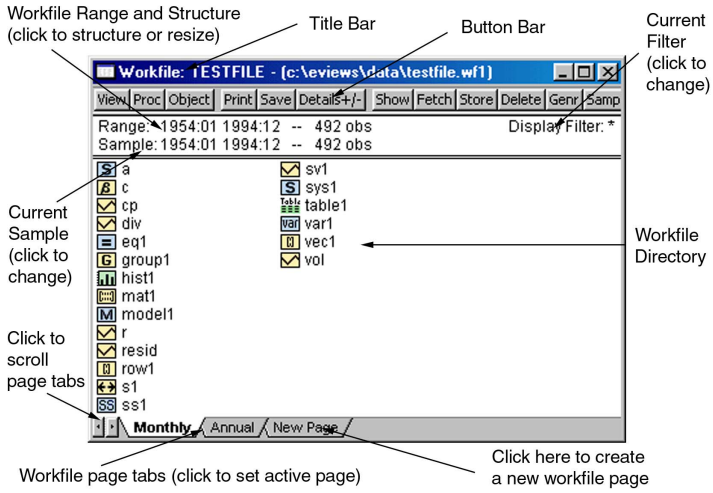
EViews will open the foreign dataset, validate the type, create an unstructured workfile, and read the selected data. When the procedure is completed, EViews will display an untitled group containing the series, and will display relevant information in the status line. In this example, EViews will report that after applying the observation filter it has retained 636 of the 1534 observations in the original dataset.

The Workfile Window

Probably the most important windows in EViews are those for workfiles. Since open workfiles contain the EViews objects that you are working with, it is the workfile window that provides you with access to all of your data. Roughly speaking, the workfile window provides you with a directory for the objects in a given workfile or workfile page. When open, the workfile window also provides you with access to tools for working with workfiles and their pages.

Workfile Directory Display

The standard workfile window view will look something like this:



In the title bar of the workfile window you will see the "**Workfile**" designation followed by the workfile name. If the workfile has been saved to disk, you will see the name and the full disk path. Here, the name of the workfile is "TESTFILE", and it is located in the "C:\EVIEWS\DATA" directory on disk. If the workfile has not been saved, it will be designated "UNTITLED".

Just below the titlebar is a button bar that provides you with easy access to useful workfile operations. Note that the buttons are simply shortcuts to items that may be accessed from the main EViews menu. For example, the clicking on the **Fetch** button is equivalent to selecting **Object/Fetch from DB...** from the main menu.

Below the toolbar are two lines of status information where EViews displays the *range* (and optionally, the structure) of the workfile, the current *sample* of the workfile (the range of observations that are to be used in calculations and statistical operations), and the display filter (rule used in choosing a subset of objects to display in the workfile window). You may change the range, sample, and filter by double clicking on these labels and entering the relevant information in the dialog boxes.

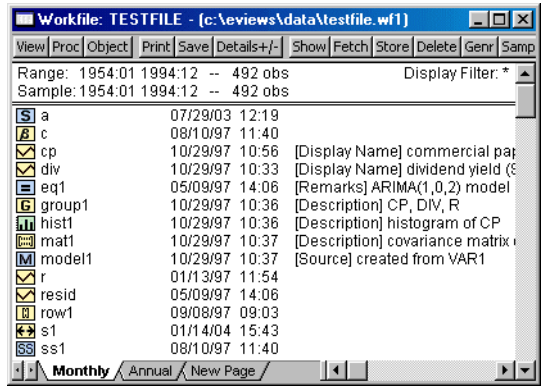
Lastly, in the main portion of the window, you will see the contents of your workfile page in the workfile directory. In normal display mode, all named objects are listed in the directory, sorted by name, with an icon showing the object type. The different types of objects and their icons are described in detail in "Object Types" on page 75 of the *Command and Programming Reference*. You may also show a subset of the objects in your workfile page, as described below.

It is worth keeping in mind that the workfile window is a specific example of an object window. Object windows are discussed in "The Object Window" on page 78.

Workfile Directory Display Options

You may choose **View/Name Display…** in the workfile toolbar to specify whether EViews should use upper or lower case letters when it displays the workfile directory. The default is lower case.

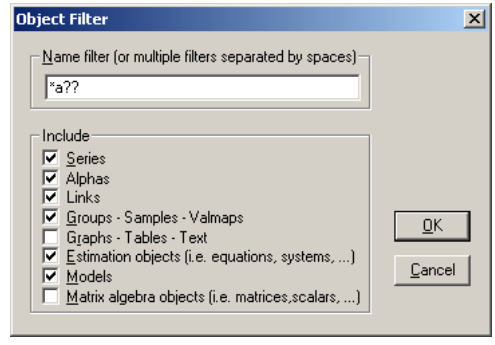
You can change the default workfile display to show additional information about your objects. If you select **View/Details + /–**, or click on the **Details + /-** button on the toolbar, EViews will toggle between the standard workfile display format, and a display which provides additional information about the date the object was created or updated, as well as the label information that you may have attached to the object.


Workfile: TESTFILE - (c:\eviews\data\testfile.wf1)
View Proc Object Print Save Details+/- Show Fetch Store Delete Genr Samp
Range: 1954:01 1994:12 -- 492 obs
Sample: 1954:01 1994:12 -- 492 obs
Display Filter: *
a 07/29/03 12:19
c 08/10/97 11:40
cp 10/29/97 10:56 [Display Name] commercial pap
div 10/29/97 10:33 [Display Name] dividend yield (9
eq1 05/09/97 14:06 [Remarks] ARIMA(1,0,2) model
group1 10/29/97 10:36 [Description] CP, DIV, R
hist1 10/29/97 10:36 [Description] histogram of CP
mat1 10/29/97 10:37 [Description] covariance matrix
model1 10/29/97 10:37 [Source] created from VAR1
r 01/13/97 11:54
resid 05/09/97 14:06
row1 09/08/97 09:03
s1 01/14/04 15:43
ss1 08/10/97 11:40
Monthly Annual New Page

Filtering the Workfile Directory Display

When working with workfiles containing a large number of objects, it may become difficult to locate specific objects in the workfile directory display. You can solve this problem by using the workfile display filter to instruct EViews to display only a subset of objects in the workfile window. This subset can be defined on the basis of object name as well as object type.

Select **View/Display Filter…** or double click on the Filter description in the workfile window. The following dialog box will appear:



There are two parts to this dialog. In the edit field (blank space) of this dialog, you may place one or several name descriptions that include the standard wildcard characters: "*" (match any number of characters) and "?" (match any single character). Below the edit field are a series of check boxes corresponding to various types of EViews

objects. EViews will display only objects of the specified types whose names match those in the edit field list.

The default string is "*", which will display all objects of the specified types. However, if you enter the string:

```
x*
```

only objects with names beginning with X will be displayed in the workfile window. Entering:

```
x?y
```

displays all objects that begin with the letter X, followed by any single character and then ending with the letter Y. If you enter:

```
x* y* *z
```

all objects with names beginning with X or Y and all objects with names ending in Z will be displayed. Similarly, the more complicated expression:

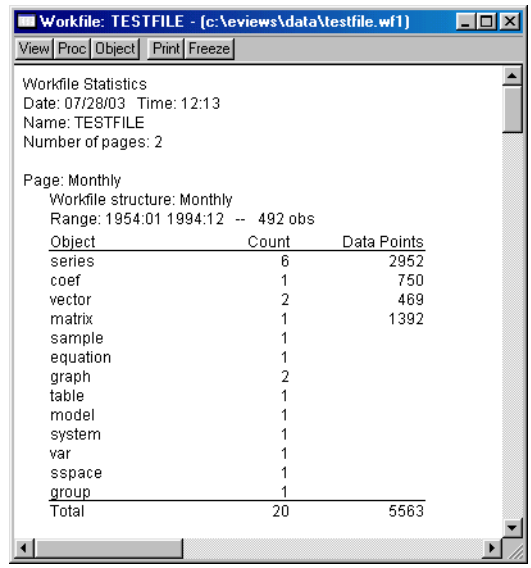
```
??y* *z*
```

tells EViews to display all objects that begin with any two characters followed by a Y and any or no characters, and all objects that contain the letter Z. Wildcards may also be used in more general settings—a complete description of the use of wildcards in EViews is provided in Appendix B, "Wildcards", on page 945.

When you specify a display filter, the Filter description in the workfile window changes to reflect your request. EViews always displays the current string used in matching names. Additionally, if you have chosen to display a subset of EViews object types, a "–" will be displayed in the **Display Filter** description at the top of the workfile window.

Workfile Summary View

In place of the directory display, you can display a summary view for your workfile. If you select this view, the display will change to provide a description of the current workfile structure, along with a list of the types and numbers of the various objects in each of the pages of the workfile.



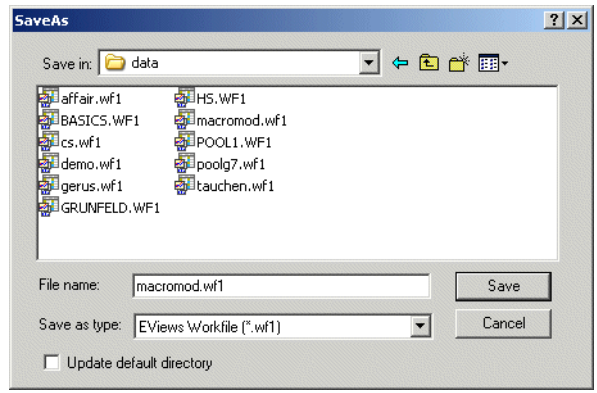
To select the summary view, click on **View/Statistics** in the main workfile menu or toolbar. Here we see the display for a first page of a two page workfile.

To return to the directory display view, select **View/Workfile Directory**.

Saving a Workfile

You should name and save your workfile for future use. Push the **Save** button on the workfile toolbar to save a copy of the workfile on disk. You can also save a file using the **File/Save As…** or **File/Save…** choices from the main menu. EViews will display the Windows common file dialog.

You can specify the target directory in the upper file menu labeled **Save in**. You can navigate between directories in the standard Windows fashion—click once on the down arrow to access a directory tree; double clicking on a directory name in the display area gives you a list of all the files and subdirectories in that directory. Once you have worked your way to the right



directory, type the name you want to give the workfile in the **File name** field and push the **Save** button.

Alternatively, you could just type the full Windows path information and name in the **File name** edit field.

In most cases, you will save your data as an EViews workfile. By default, EViews will save your data in this format, using the specified name and the extension ".WF1". You may, of course, choose to save the data in your workfile in a foreign data format by selecting a different format in the combo box. We explore the subject of saving foreign formats below in "Exporting from a Workfile" on page 259.

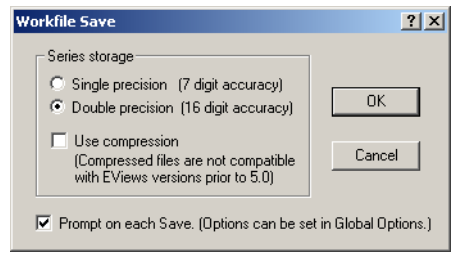
Saving Updated Workfiles

You may save modified or updated versions of your named workfile using the **Save** button on the workfile toolbar, or by selecting **File/Save...** from the main menu. Selecting **Save** will update the existing workfile stored on disk. You may also use **File/Save As...** to save the workfile with a new name. If the file you save to already exists, EViews will ask you whether you want to update the version on disk.

When you overwrite a workfile on disk, EViews will usually keep a backup copy of the overwritten file. The backup copy will have the same name as the file, but with the first character in the extension changed to ~. For example, if you have a workfile named MYDATA.WF1, the backup file will be named MYDATA.~F1. The existence of these backup files will prove useful if you accidentally overwrite or delete the current version of the workfile file, or if the current version becomes damaged.

If you wish to turn on or off the creation of these backup copies you should set the desired global options by selecting **Options/Workfile Storage Defaults...**, and selecting the desired settings.

Workfile Save Options


Workfile Save
Series storage
Single precision (7 digit accuracy)
Double precision (16 digit accuracy)
Use compression
(Compressed files are not compatible with EViews versions prior to 5.0)
OK
Cancel
Prompt on each Save. (Options can be set in Global Options.)

By default, when you click on the **Save** button, EViews will display a dialog showing the current global default options for storing the data in your workfile.

Your first choice is whether to save your series data in either **Single precision** or **Double precision**. Single precision will create smaller files on disk, but saves the data with fewer digits of accuracy (7 versus 16).

You may also choose to save your data in compressed or non-compressed form. If you select **Use compression**, EViews will analyze the contents of your series, choose an opti-

mal (lossless) storage precision for each series, and will apply compression algorithms, all to reduce the size of the workfile on disk. The storage savings may be considerable, especially for large datasets containing lots of integer and 0, 1 variables. We caution however, that a compressed workfile is not backward compatible, and will not be readable by versions of EViews prior to 5.0.

Lastly, there is a checkbox for showing the options dialog on each save operation. By default, the dialog will be displayed every time you save a workfile. Unchecking the **Prompt on each Save** option instructs EViews to hide this dialog on subsequent saves. If you later wish to change the save settings or wish to display the dialog on saves, you must update your global settings by selecting **Options/Workfile Default Storage Options...** from the main EViews menu.

Note that, with the exception of compressed workfiles, workfiles saved in EViews 5 may be read by previous versions of EViews. Objects such as valmaps or alpha series that are not supported by previous versions will, however, be dropped when read by earlier versions of EViews. You should take great caution when saving workfiles using older versions of EViews as you will lose these deleted objects (see "Workfile Compatibility" on page 5).

Note also that only the first page of a multi-page workfile will be read by previous versions; all other pages will be dropped. You may save individual pages of a multi-page workfile to separate workfiles so that they may be read by previous versions; see "Saving a Workfile Page" on page 71.

Loading a Workfile

You can use **File/Open/EViews Workfile...** to load into memory a previously saved workfile. You will typically save a workfile containing all of your data and results at the end of the day, and later load the workfile to pick up where you left off.

When you select **File/Open/EViews Workfile...** you will see a standard Windows file dialog. Simply navigate to the appropriate directory and double click on the name of the workfile to load it into RAM. The workfile window will open and all of the objects in the workfile will immediately be available.

For convenience, EViews keeps a record of the most recently used files at the bottom of the **File** menu. Select an entry and it will be opened in EViews.

Version 5 of EViews can read workfiles from all previous versions of EViews. Due to changes in the program, however, some objects may be modified when they are read into EViews 5.

Multi-page Workfiles

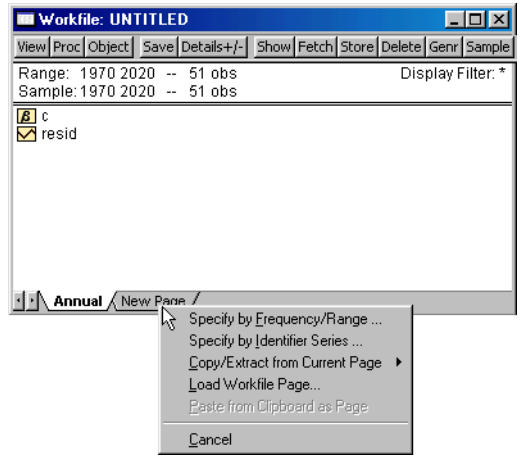
While a great many of your workfiles will probably contain a single page, you may find it useful to organize your data into multiple workfile pages. Multi-page workfiles are primarily designed for situations in which you must work with multiple datasets.

For example, you may have both quarterly and monthly data that you wish to analyze. The multi-page workfile allows you to hold both sets of data in their native frequency, and to perform automatic frequency conversion as necessary. Organizing your data in this fashion allows you to switch instantly between performing your analysis at the monthly and the quarterly level.

Likewise, you may have a panel dataset on individuals that you wish to use along with a cross-sectional dataset on state level variables. By creating a workfile with a separate page for the individual level data, and a separate page for the state level data, you can move back and forth between the individual and the state level analyses, or you can link data between the two to perform dynamic match merging.

Creating a Workfile Page

There are several ways to create a new workfile page.



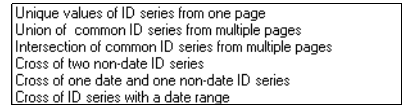
Creating a Page by Describing its Structure

First, you may describe the structure of the workfile page. This method follows the approach outlined in "Creating a Workfile by Describing its Structure" on page 51. Simply call up the new page menu by clicking on the tab labeled **New Page** and selecting **Specify by Frequency/Range...**, and EViews will display the familiar **Workfile Create** dialog. Simply describe the structure of your workfile page as you would for a new workfile, and enter **OK**.

EViews will create a new workfile page with the specified structure and the new page will be given a default name and designated as the active workfile page. The default name will be constructed from the next available name for the given workfile structure. For example, if you create a regular frequency annual page, EViews will attempt to name the page ANNUAL, ANNUAL1, and so forth. The active page is noted visually by the tab selection at the bottom of the workfile window. With the exception of a few page-specific operations, you may generally treat the active page as if it were a standard workfile.

Creating a Workfile Page Using Identifiers

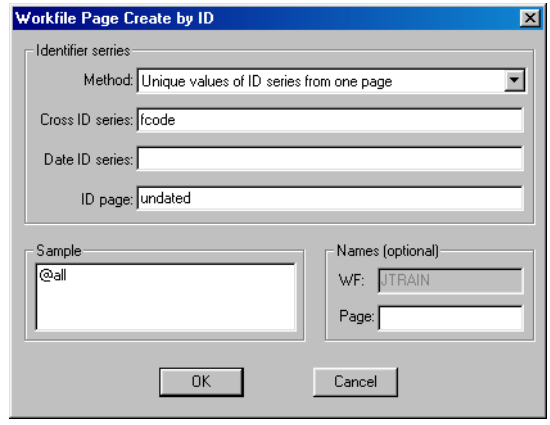
The second approach creates a new page using the unique values of one or more identifier series. Click on the **New Page** tab and select **Specify by Identifier Series...** EViews will open a dialog for creating a new page using one or more identifier series.


Unique values of ID series from one page
Union of common ID series from multiple pages
Intersection of common ID series from multiple pages
Cross of two non-date ID series
Cross of one date and one non-date ID series
Cross of ID series with a date range

At the top of the dialog is a combo box labeled **Method** that you may use to select between the various ways of using identifiers to specify a new page. You may choose between creating the page using: (1) the unique ID values from the current workfile page, (2) the union of unique ID values from multiple pages, (3) the intersection of unique ID values from mulitple pages, (4) and (5) the cross of the unique values of two ID series, (6) the cross of a single ID series with a date range.

As you change the selected method, the dialog will change to provide you with different options for specifying identifiers.

Unique values of ID series from one page



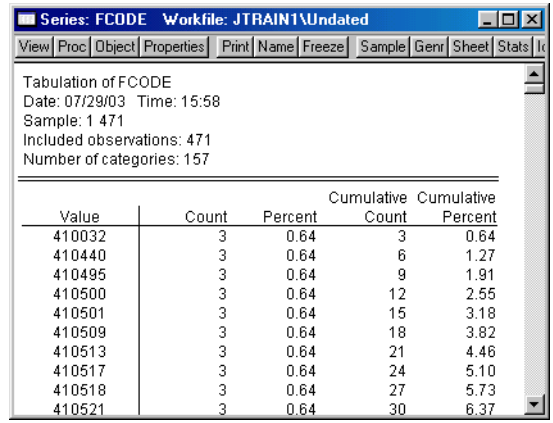
The easiest way to create a new page from identifiers is to use the unique values in one or more series in the current workfile page.

If you select **Unique values of ID series from one page** in the **Method** combo, EViews will prompt you for one or more identifier series which you should enter in the **Cross-section ID series** and **Date series** edit fields.

EViews will take the set of series and will identify the unique values in the specified **Sample**. Note that when multiple identifiers are specified, the unique values are defined over the values in the set of ID series, not over each individual series.

The new page will contain identifier series containing the unique values, and EViews will structure the workfile using this information. If **Date ID** series were provided in the original dialog, EViews will restructure the result as a dated workfile page.

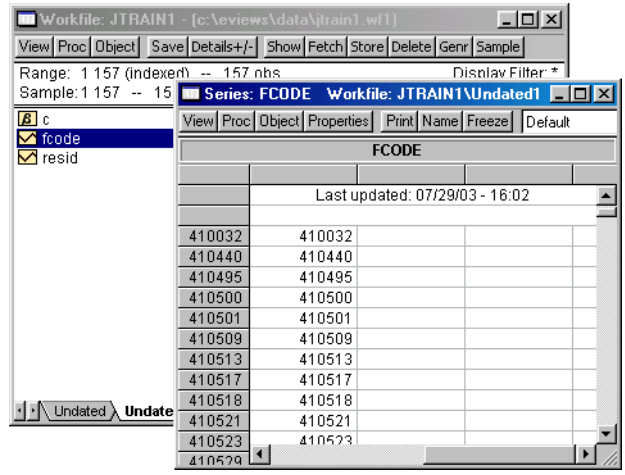
Suppose, for example, that we begin with a workfile page UNDATED that contains 471 observations on 157 firms observed for 3 years. There is a series FCODE identifying the firm, and a series YEAR representing the year.


Series: FCODE Workfile: JTRAIN1\Undated
View Proc Object Properties Print Name Freeze Sample Genr Sheet Stats
Tabulation of FCODE
Date: 07/29/03 Time: 15:58
Sample: 1 471
Included observations: 471
Number of categories: 157
Value Count Percent Cumulative Count Cumulative Percent
410032 3 0.64 3 0.64
410440 3 0.64 6 1.27
410495 3 0.64 9 1.91
410500 3 0.64 12 2.55
410501 3 0.64 15 3.18
410509 3 0.64 18 3.82
410513 3 0.64 21 4.46
410517 3 0.64 24 5.10
410518 3 0.64 27 5.73
410521 3 0.64 30 6.37

We first wish to create a new workfile page containing 157 observations representing the unique values of FCODE. Simply enter FCODE in the **Cross-section ID series,** set the sample to "@ALL", name the new page "UNDATED1", and click on **OK.**

EViews will create a new structured (undated - with identifier series) workfile page UNDATED1 containing 157 observations. The new page will contain a series FCODE with the 157 unique values found in the original series FCODE, and the workfile will be structured using this series.

Similarly, we may choose to create a new page using the series YEAR, which identifies the year that the firm was observed. There are three distinct values for YEAR in the original workfile page ("1987", "1988", "1989"). Click on the Click on the **New Page** tab and select **Specify by Identifier Series...** from the menu, and **Unique values of ID series from one page** in the **Method** combo. Enter "YEAR" in the **Date ID** series field, and click on **OK** to create a new annual page with range 1987–1989. Note that EViews will structures the result as a dated workfile page.



Union of common ID series from multiple pages

In some cases, you may wish to create your new page using unique ID values taken from more than one workfile page.

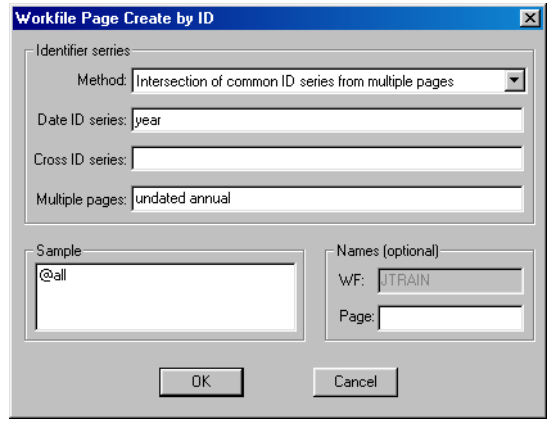
If you select **Union of commmon ID series from multiple pages**, EViews will find, for each source page, a set of unique ID values, and will create the new workfile page using the union of these values. Simply enter the list of identifiers in the **Cross-section ID series** and **Date series** and edit fields, and a list of pages in which the common identifiers may be found. When you click on **OK**, EViews will first make certain that each of the identifier series is found in each page, then will create the new workfile page using the union of the observed ID values.

We may extend our earlier example where there are three distinct values for YEAR in the original page ("1987", "1988", "1989"). To make things more interesting, suppose there is a second page in the workfile, ANNUAL, containing annual data for the years 1985–1988 and that this page contains also contains a series YEAR with those values ("1985", "1986", "1987", "1988").

Since we want to exploit the fact that YEAR contains date information, we create a page using the union of IDs by selecting **Union of common ID series from multiple pages**, entering YEAR in the **Date series** field, and then entering "UNDATED" and "ANNUAL" in the page field. When you click on **OK**, EViews will create a 5 observation, regular annual frequency workfile page for 1987–1989, formed by taking the union of the unique values in the YEAR series in the UNDATED panel page, and the YEAR series in the ANNUAL page.

Intersection of common ID series from multiple pages

In other cases, you may wish to create your new page using common unique ID values taken from more than one workfile page. If you select **Intersection of commmon ID series from multiple pages**, EViews willtake the specified set of series and will identify the unique values in the specified **Sample**. The intersection of these sets of unique values across the pages will then be used to create a new workfile page.



In our extended YEAR example, we have two pages: UNDATED, with 471 observations and 3 distinct YEAR values ("1987", "1988", and "1989"); and the ANNUAL workfile page containing annual data for four years from 1985–1988, with corresponding values for the series YEAR.

Suppose that we enter YEAR in the **Date ID** field, and tell EViews to examine the intersection of values in the **Multiple pages** UNDATED and ANNUAL. EViews will create a new workfile page containing the intersection of the unique

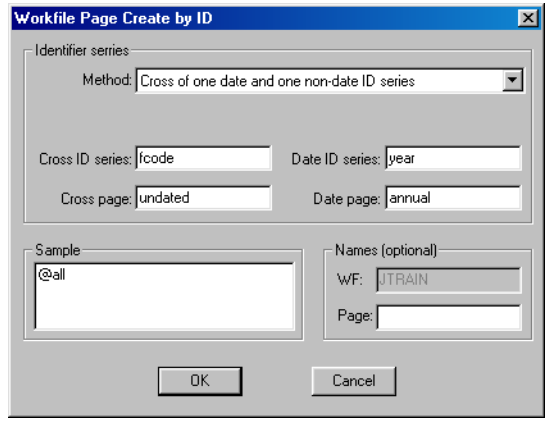
values of the YEAR series across pages ("1987", "1988"). Since YEAR was specified as a date ID, the page will be structured as a dated annual page.

Cross of two ID series

There are two choices if you wish to create a page by taking the cross of the unique values from two ID series: **Cross of two non-date ID series** creates an undated panel page using the unique values of the two identifiers, while **Cross of one date and one non-date ID series** uses the additional specification of a date ID to allow for the structuring of a dated panel page.

Suppose for example, that you wish to create a page by crossing the 187 unique FCODE values in the UNDATED page with the 4 unique YEAR values in the ANNUAL page ("1985", "1986", "1987", "1988"). Since the YEAR values may be used to create a dated panel, we select **Cross of one date and one non-date ID** from our **Method** combo.

Since we wish to use YEAR to date structure our result, we enter "FCODE" and "UNDATED" in the **Cross ID series** and **Cross page** fields, and we enter "YEAR" and "ANNUAL" in the **Date ID series** and **Date page** fields.



When you click on **OK**, EViews will create a new page by crossing the unique values of the two ID series. The resulting workfile will be an annual dated panel for 1985–1988, with FCODE as the cross-section identifer.

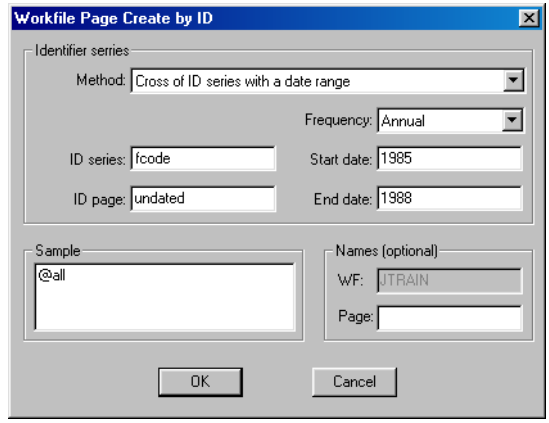
It is worth noting that had we had entered the same information in the **Cross of two non-date ID** dialog, the result would be an undated panel with two identifier series.

Cross of ID Series with a date range

In our example of crossing a date ID series with a non-date ID, we were fortunate to have an annual page to use in construting the date ID. In some cases, the dated page may not be immediately available, and will have to be created prior to performing the crossing operation.

In cases where the page is not available, but where we wish to cross our non-date ID series with a regular frequency range, we may skip the intermediate page creation by selecting the **Cross of ID series with a date range** method.

Here, instead of specifying a date ID series and page, we need only specify a page frequency, start, and end dates. In this example, the resulting annual panel page is identical to the page specified by crossing FCODE with the YEAR series from the ANNUAL page.



While specifying a frequency and range is more convenient than specifying a date ID and page, this method is obviously more restrictive since it does not allow for irregular dated data. In these latter cases, you must explicitly specify your date ID series and page.

Creating a Page by Copying the Current Page

You may also create a new workfile page by copying data from the current page. Click on **New Page** or click on **Proc** in the main workfile menu, and select **Copy/Extract from Current Page** and either **By Link to New Page...** or **By Value to New Page or Workfile...**. EViews will open a dialog prompting you to specify the objects and data that you wish to copy to a new page. See "Copying from a Workfile" on page 238 for a complete discussion.

Creating a Page by Loading a Workfile or Data Source

The next method for creating a new page is to load an existing workfile or data source. Call up the new page menu by clicking on **New Page** and selecting **Load Workfile Page...** or by selecting **Proc/Load Workfile Page...** from the main workfile menu. EViews will present you with the **File Open** dialog, prompting you to select your file.

If you select an existing EViews workfile, EViews will add a page corresponding to each page in the source workfile. If you load a workfile with a single page named QUARTERLY, EViews will attempt to load the entire workfile in the new page. If your workfile contains multiple pages, each page of the workfile will be loaded into a new and separate page. The active page will be the newest page.

If you select a foreign data source as described in "Creating a Workfile by Reading from a Foreign Data Source" on page 53, EViews will load the data into a single newly created page in the workfile. This method is exactly the same as that used when creating a new workfile except that the results are placed in a new workfile page.

Creating a Page by Pasting from the Clipboard

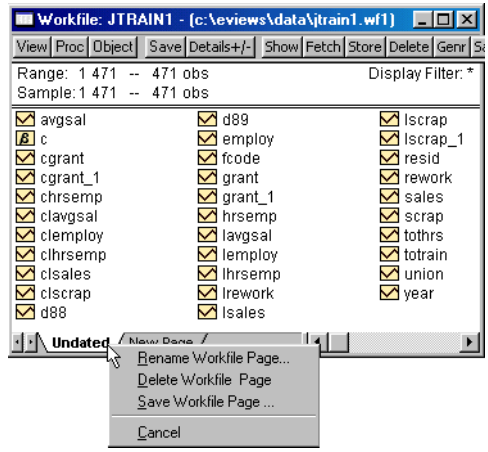
You may create a new workfile page by pasting the contents of the Windows Clipboard. This method is particularly useful for copying and pasting data from another application such as Microsoft Word, Excel, or your favorite web browser.

Simply copy the data you wish to use in creating your page, then click on **New Page** and select **Paste from Clipboard as Page**. EViews will first analyze the contents of the clipboard. EViews then creates a page to hold the data and then will read the data into series in the page.

Note that while EViews can correctly analyze a wide range of data representations, the results may not be as expected in more complex settings.

Working With Workfile Pages

While workfile pages may generally be thought of simply as workfiles, there are certain operations that are page-specific or fundamental to multi-page workfiles.



Setting the Active Workfile Page

To select the active workfile page, simply click on the visible tab for the desired page in the workfile window. The active page is noted visually by the tab selection at the bottom of the workfile window.

If the desired page is not visible, you may click on the small right and left arrows in the bottom left-hand corner of the workfile window to scroll the page tab display until the desired page is visible, then click on the tab.

You should note that it is possible to hide existing page tabs. If a page appears to be missing, for example if **New Page** is the only visible tab, the remaining tabs are probably hidden. You should click on the left arrow located in the bottom right of the workfile window until your page tabs are visible.

Renaming a Workfile Page

EViews will give your workfile pages a default name corresponding to the workfile structure. You may wish to rename these pages to something more informative. Simply click on the tab for the page you wish to rename and right-mouse-button click to open the workfile page menu. Select **Rename Workfile Page...** from the menu and enter the page name.

Alternatively, you may select **Proc/Rename Current Page...** from the main workfile menu to call up the dialog.

Workfile page names must satisfy the same naming restrictions as EViews objects. Notably, the page names must not contain spaces or other delimiters.

Deleting a Workfile Page

To delete a workfile page, right mouse click on the page tab and select **Delete Workfile Page**, or with the page active, click on the **Proc** menu and select **Delete Current Page**.

Saving a Workfile Page

If you wish to save the active workfile page as an individual workfile click on the page tab, right mouse click to open the workfile page menu and select **Save Workfile Page...** to open the **SaveAs** dialog. Alternatively, you may select **Proc/Save Current Page...** from the main workfile menu to access the dialog.

Saving a page as an individual workfile is quite useful when you wish to load a single page into several workfiles, or if you wish to use the page in a previous version of EViews. Once saved on disk, it is the same as any other single-page EViews workfile.

Addendum: File Dialog Features

There are additional features in the file open and save dialogs which you may find useful.

Set Default Directory

All EViews file dialogs begin with a display of the contents of the *default directory*. You can always identify the default directory from the listing on the EViews status line. The default directory is set initially to be the directory containing the EViews program, but it can be changed at any time.

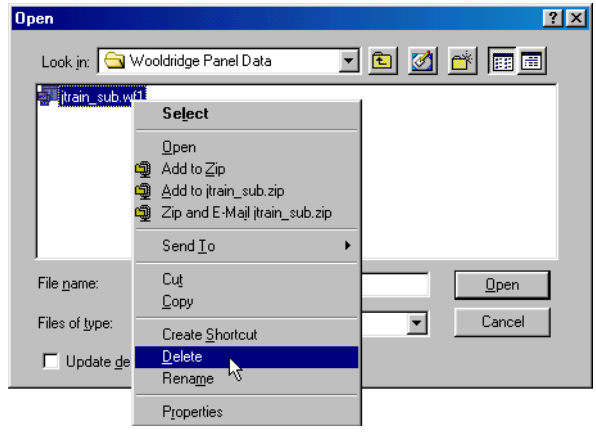
You can change the default directory by using the **File/Open...** or the **File/Save As...** menu items, navigating to the new directory, and checking the **Update Default Directory** box in the dialog. If you then open or save a workfile, the default directory will change to the one you have selected. The default directory may also be set from the **Options/File locations...** dialog. See "File Locations" on page 938.

An alternative method for changing the default EViews directory is to use the `cd` command. Simply enter "CD" followed by the directory name in the command window (see `cd` (p. 234) of the *Command and Programming Reference* for details).

File Operations

Since EViews uses a variant of the Windows common file dialog for all open and save operations, you may use the dialog to perform routine file operations such as renaming, copying, moving, and deleting files.

For example, to delete a file, click once of the file name to select the file, then right click once to call up the menu, and select **Delete**. Likewise, you may select a file, right-mouse click, and perform various file operations such as **Copy** or **Rename**.



Chapter 4. Object Basics

At the heart of the EViews design is the concept of an object. In brief, objects are collections of related information and operations that are bundled together into an easy-to-use unit. Virtually all of your work in EViews will involve using and manipulating various objects.

EViews holds all of its objects in object containers. You can think of object containers as filing cabinets or organizers for the various objects with which you are working. The most important object container in EViews is the workfile, which is described in Chapter 3, "Workfile Basics", beginning on page 49.

The remainder of this chapter describes basic techniques for working with objects in a workfile. While you may at first find the idea of objects to be a bit foreign, the basic concepts are easy to master and will form the foundation for your work in EViews. But don't feel that you have to understand all of the concepts the first time through. If you wish, you can begin working with EViews immediately, developing an intuitive understanding of objects and workfiles as you go.

Subsequent chapters will provide a more detailed description of working with the various types of objects and other types of object containers.

What is an Object?

Information in EViews is stored in *objects*. Each object consists of a collection of information related to a particular area of analysis. For example, a *series object* is a collection of information related to a set of observations on a particular variable. An *equation object* is a collection of information related to the relationship between a collection of variables.

Note that an object need not contain only one type of information. For example, an estimated equation object contains not only the coefficients obtained from estimation of the equation, but also a description of the specification, the variance-covariance matrix of the coefficient estimates, and a variety of statistics associated with the estimates.

Associated with each type of object is a set of views and procedures which can be used with the information contained in the object. This association of views and procedures with the type of data contained in the object is what we term the *object oriented* design of EViews.

The object oriented design simplifies your work in EViews by organizing information as you work. For example, since an equation object contains all of the information relevant to an estimated relationship, you can move freely between a variety of equation specifications simply by working with different equation objects. You can examine results, perform

hypothesis and specification tests, or generate forecasts at any time. Managing your work is simplified since only a single object is used to work with an entire collection of data and results.

This brief discussion provides only the barest introduction to the use of objects. The remainder of this section will provide a more general description of EViews objects. Subsequent chapters will discuss series, equations, and other object types in considerable detail.

Object Data

Each object contains various types of information. For example, series, matrix, vector, and scalar objects, all contain mostly numeric information. In contrast, equations and systems contain complete information about the specification of the equation or system, and the estimation results, as well as references to the underlying data used to construct the estimates. Graphs and tables contain numeric, text, and formatting information.

Since objects contain various kinds of data, you will want to work with different objects in different ways. For example, you might wish to compute summary statistics for the observations in a series, or you may want to perform forecasts based upon the results of an equation. EViews understands these differences and provides you with custom tools, called views and procedures, for working with an object's data.

Object Views

There is more than one way to examine the data in an object. Views are tabular and graphical windows that provide various ways of looking at the data in an object.

For example, a series object has a spreadsheet view, which shows the raw data, a line graph view, a bar graph view, a histogram-and-statistics view, and a correlogram view. Other views of a series include distributional plots, QQ-plots, and kernel density plots. Series views also allow you to compute simple hypothesis tests and statistics for various subgroups of your sample.

An equation object has a representation view showing the equation specification, an output view containing estimation results, an actual-fitted-residual view containing plots of fitted values and residuals, a covariance view containing the estimated coefficient covariance matrix, and various views for specification and parameter tests.

Views of an object are displayed in the object's window. Only one window can be opened for each object and each window displays only a single view of the object at a time. You can change views of an object using the **View** menu located in the object window's toolbar or the EViews main menu.

Perhaps the most important thing to remember about views is that views normally do not change data outside the object. Indeed, in most cases, changing views only changes the display format for the data, and not the data in the object itself.

Object Procedures

Most EViews objects also have *procedures*, or *procs*. Like views, procedures often display tables or graphs in the object's window. Unlike views, however, procedures alter data, either in the object itself or in another object.

Many procedures create new objects. For example, a series object contains procedures for smoothing or seasonally adjusting time series data and creating a new series containing the smoothed or adjusted data. Equation objects contain procedures for generating new series containing the residuals, fitted values, or forecasts from the estimated equation.

You select procedures from the **Proc** menu on the object's toolbar or from the EViews main menu.

Object Types

The most common objects in EViews are series and equation objects. There are, however, a number of different types of objects, each of which serves a unique function. Most objects are represented by a unique icon which is displayed in the object container (workfile or database) window. The basic object icons are given by:

Alpha	Model	Sym
Coefficient Vector	Pool	System
Equation	Rowvector	Table
Graph	Sample	Text
Group	Scalar	Valmap
Logl	Series	VAR
Matrix	Sspace	Vector

Despite the fact that they are also objects, object containers do not have icons since they cannot be placed in other object containers—thus, workfiles and databases do not have icons since they cannot be placed in other workfiles or databases.

Note also that there are special icons that correspond to special versions of the objects:

Auto-updating Series	
Group data and definitions (in databases)	

Undefined Link

If you set a series object to be auto-updating (see "Auto-Updating Series" on page 149), EViews will use the special icon to indicate that the series depends upon a formula. In contrast, an auto-updating alpha series (which we imagine to be less common) uses the original alpha icon, with an orange color to indicate the presence of a formula.

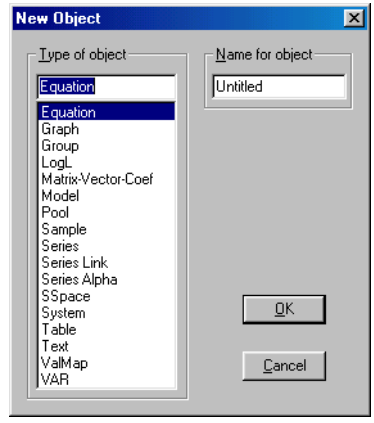
When group data are stored in databases, you will be given the option of storing the group definition (list of series names) alone, or both the group definition and the series contained in the group (see "Store, Fetch, and Copy of Group Objects" on page 272). If the latter are stored, the standard group icon will be modified, with the " + " indicating the additional presence of the series data.

Lastly, a link object (see "Series Links" on page 177), is always in one of three states, depending upon the definition contained in the link. If the link is to a numeric source series, the link object be displayed using a series icon, since it may be used as though it were an ordinary series, with a distinctive pink color used to indicate that the object depends on linked data. If the link is to an alpha source series, the link will show up as an alpha series icon, again in pink. If, however, the link object is unable to locate the source series, EViews will display the "?" icon indicating that the series type is unknown.

Basic Object Operations

Creating Objects

To create an object, you must first make certain that you have an open workfile container and that its window is active. Next, select **Object/New Object...** from the main menu. Until you have created or loaded a workfile, this selection is unavailable. After you click on the **Object/New Object...** menu entry, you will see the **New Object** dialog box.

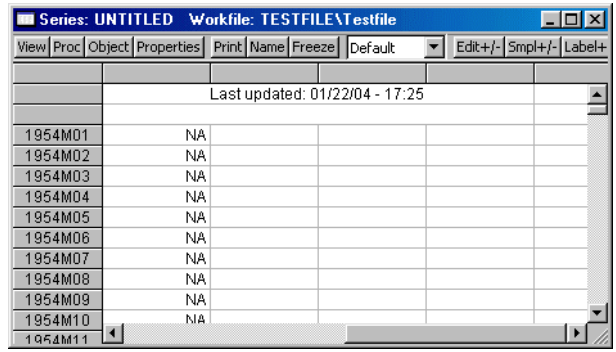


You can click on the type of object you want, optionally provide a name for the object, and then click on **OK**. For some object types, a second dialog box will open prompting you to describe your object in more detail. For most objects, however, the object window will open immediately.

For example, if you select **Equation**, you will see a dialog box prompting you for additional information. Alternatively, if you click on **Series** and then select **OK**, you will see an object window (series window) displaying the spreadsheet view of an UNTITLED series.

We will discuss object windows in greater detail in "The Object Window" on page 78.

Objects can also be created by applying procedures to other objects or by freezing an object view (see "Freezing Objects" on page 83).



Selecting Objects

Creating a new object will not always be necessary. Instead, you may want to work with an existing object. One of the fundamental operations in EViews is *selecting* one or more objects from the workfile directory.

The easiest way to select objects is to point-and-click, using the standard Windows conventions for selecting contiguous or multiple items if necessary ("Selecting and Opening Items" on page 20). Keep in mind that if you are selecting a large number of items, you may find it useful to use the display filter before beginning to select items.

In addition, the **View** button in the workfile toolbar provides convenient selection shortcuts:

- **Select All** selects all of the objects in the workfile with the exception of the C coefficient vector and the RESID series.
- **Deselect All** eliminates any existing selections.

Note that all of the selected objects will be highlighted.

Opening Objects

Once you have selected your object or objects, you will want to open your selection, or create a new object containing the selected objects. You can do so by double clicking anywhere in the highlighted area.

If you double click on a single selected object, you will open an object window.

If you select multiple graphs or series and double click, a pop-up menu appears, giving you the option of creating and opening new objects (group, equation, VAR, graph) or displaying each of the selected objects in its own window.

Note that if you select multiple graphs and double click or select **View/Open as One Window**, all of the graphs will be merged into a single graph and displayed in a single window.

Other multiple item selections are not valid, and will either issue an error or will simply not respond when you double click.

When you open an object, EViews will display the current view. In general, the current view of an object is the view that was displayed the last time the object was opened (if an object has never been opened, EViews will use a default view). The exception to this general rule is for those views that require significant computational time. In this latter case, the current view will revert to the default.

Showing Objects

An alternative method of selecting and opening objects is to "show" the item. Click on the **Show** button on the toolbar, or select **Quick/Show…** from the menu and type in the object name or names.

Showing an object works exactly as if you first selected the object or objects, and then opened your selection. If you enter a single object name in the dialog box, EViews will open the object as if you double clicked on the object name. If you enter multiple names, EViews will always open a single window to display results, creating a new object if necessary.

The **Show** button can also be used to display functions of series, also known as auto-series. All of the rules for auto-series that are outlined in "Database Auto-Series" on page 274, will apply.

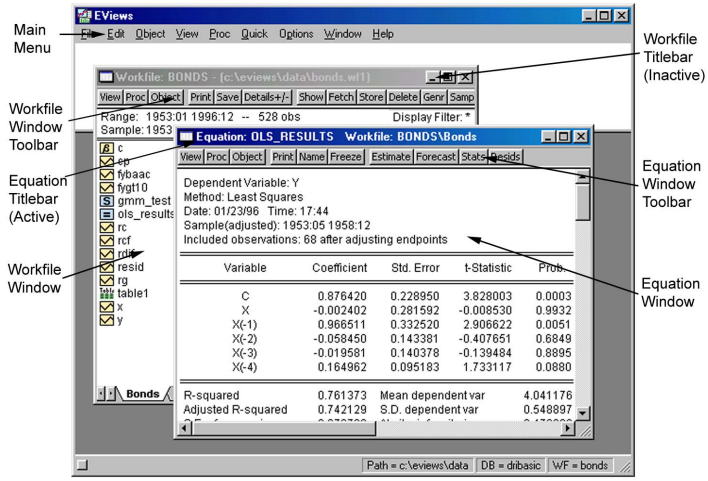
The Object Window

We have been using the term *object window* somewhat loosely in the previous discussion of the process of creating and opening objects. Object windows are the windows that are displayed when you open an object or object container. An object's window will contain either a view of the object, or the results of an object procedure.

One of the more important features of EViews is that you can display object windows for a number of items at the same time. Managing these object windows is similar to the task of managing pieces of paper on your desk.

Components of the Object Window

Let's look again at a typical object window:


Main Menu
Workfile Window Toolbar
Equation Titlebar (Active)
Workfile Window
Workfile Titlebar (Inactive)
Equation Window Toolbar
Equation Window
EViews
File Edit Object View Proc Quick Options Window Help
Workfile: BONDS - [c:\eviews\data\bonds.wf1]
Equation: OLS_RESULTS Workfile: BONDS\Bonds
Dependent Variable: Y
Method: Least Squares
Date: 01/23/96 Time: 17:44
Sample(adjusted): 1953:05 1958:12
Included observations: 68 after adjusting endpoints
Variable Coefficient Std. Error t-Statistic Prob.
C 0.876420 0.228950 3.828003 0.0003
X -0.002402 0.281592 -0.008530 0.9932
X(-1) 0.966511 0.332520 2.906622 0.0051
X(-2) -0.058450 0.143381 -0.407651 0.6849
X(-3) -0.019581 0.140378 -0.139484 0.8895
X(-4) 0.164962 0.095183 1.733117 0.0880
R-squared 0.761373 Mean dependent var 4.041176
Adjusted R-squared 0.742129 S.D. dependent var 0.548897
Path = c:\eviews\data
DB = dribasic
WF = bonds

Here, we see the equation window for OLS_RESULTS. First, notice that this is a standard window which can be closed, resized, minimized, maximized, and scrolled both vertically and horizontally. As in other Windows applications, you can make an object window active by clicking once on the titlebar, or anywhere in its window. Making an object window active is equivalent to saying that you want to work with that object. Active windows may be identified by the darkened titlebar.

Second, note that the titlebar of the object window identifies the object type, name, and object container (in this case, the BONDS workfile or the OLS_RESULTS equation). If the object is itself an object container, the container information is replaced by directory information.

Lastly, at the top of the window there is a toolbar containing a number of buttons that provide easy access to frequently used menu items. These toolbars will vary across objects—the series object will have a different toolbar from an equation or a group or a VAR object.

There are several buttons that are found on all object toolbars:

- The **View** button lets you change the view that is displayed in the object window. The available choices will differ, depending upon the object type.
- The **Proc** button provides access to a menu of procedures that are available for the object.
- The **Object** button lets you manage your objects. You can store the object on disk, name, delete, copy, or print the object.
- The **Print** button lets you print the current view of the object (the window contents).

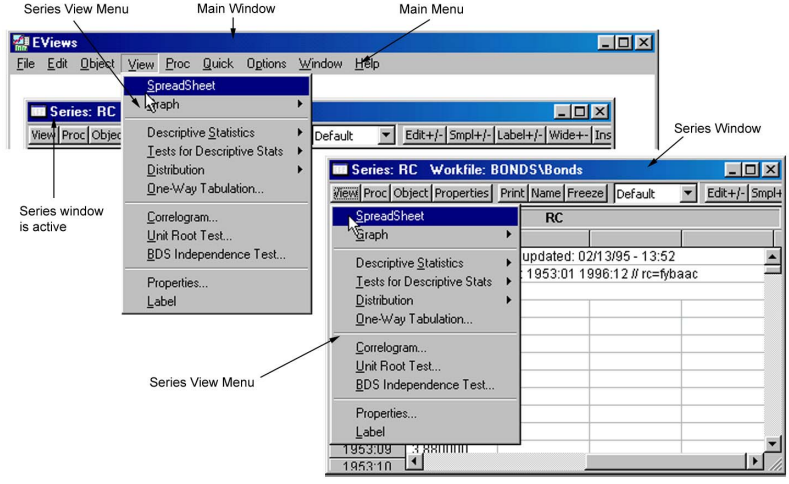
- The **Name** button allows you to name or rename the object.
- The **Freeze** button creates a new object graph, table, or text object out of the current view.

Menus and the Object Toolbar

As we have seen, the toolbar provides a shortcut to frequently accessed menu commands. There are a couple of subtle, but important, points associated with this relationship that deserve special emphasis:

- Since the toolbar simply provides a shortcut to menu items, you can always find the toolbar commands in the menus.
- This fact turns out to be quite useful if your window is not large enough to display all of the buttons on the toolbar. You can either enlarge the window so that all of the buttons are displayed, or you can access the command directly from the menu.
- The toolbar and menu *both* change with the object type. In particular, the contents of the **View** menu and the **Proc** menu will always change to reflect the type of object (series, equation, group, etc.) that is active.

The toolbars and menus differ across objects. For example, the **View** and **Proc** drop-down menus differ for every object type. When the active window is displaying a series window, the menus provide access to series views and series procedures. Alternatively, when the active window is a group window, clicking on **View** or **Proc** in the main menu provides access to the different set of items associated with group objects.



The figure above illustrates the relationship between the View toolbar button and the View menu when the series window is the active window. In the left side of the illustration, we

see a portion of the EViews *main window,* as it appears, after you click on View in the main menu (note that the RC series window is the active window). On the right, we see a depiction of the *series window* as it appears after you click on the **View** button in the series toolbar. Since the two operations are identical, the two drop-down menus are identical.

In contrast to the **View** and **Proc** menus, the **Object** menu does not, in general, vary across objects. An exception occurs, however, when an object container window (a workfile or database window) is active. In this case, clicking on **Object** in the toolbar, or selecting **Object** from the menu provides access to menu items for manipulating the objects in the container.

Working with Objects

Naming Objects

Objects may be named or unnamed. When you give an object a name, the name will appear in the directory of the workfile, and the object will be saved as part of the workfile when the workfile is saved.

You must name an object if you wish to keep its results. If you do not name an object, it will be called "UNTITLED". Unnamed objects are not saved with the workfile, so they are deleted when the workfile is closed and removed from memory.

To rename an object, first open the object window by double clicking on its icon, or by clicking on **Show** on the workfile toolbar, and entering the object name. Next, click on the **Name** button on the object window, and enter the name (up to 24 characters), and optionally, a display name to be used when labelling the object in tables and graphs. If no display name is provided, EViews will use the object name.

You can also rename an object from the workfile window by selecting **Object/Rename Selected…** and then specifying the new object name. This method saves you from first having to open the object.

The following names are reserved and cannot be used as object names: ABS, ACOS, AND, AR, ASIN, C, CON, CNORM, COEF, COS, D, DLOG, DNORM, ELSE, ENDIF, EXP, LOG, LOGIT, LPT1, LPT2, MA, NA, NOT, NRND, OR, PDL, RESID, RND, SAR, SIN, SMA, SQR, and THEN.

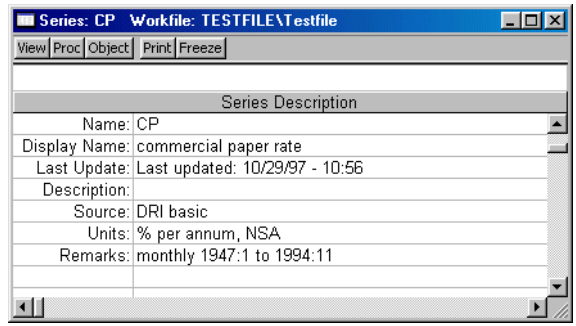
EViews accepts both capital and lower case letters in the names you give to your series and other objects, but does not distinguish between names based on case. Its messages to you will follow normal capitalization rules. For example, "SALES", "sales", and "sAles" are all the same object in EViews. For the sake of uniformity, we have written all examples of input using names in lower case, but you should feel free to use capital letters instead.

Despite the fact that names are not case sensitive, when you enter text information in an object, such as a plot legend or label information, your capitalization will be preserved.

By default, EViews allows only one untitled object of a given type (one series, one equation, etc.). If you create a new untitled object of an existing type, you will be prompted to name the original object, and if you do not provide one, EViews will replace the original untitled object with the new object. The original object will not be saved. If you prefer, you can instruct EViews to retain all untitled objects during a session but you must still name the ones you want to save with the workfile. See "Window and Font Options" on page 937.

Labeling Objects

In addition to the display name described above, EViews objects have label fields where you can provide extended annotation and commentary. To view these fields, select **View/Label** from the object window:



This is the label view of an unmodified object. By default, every time you modify the object, EViews automatically records the modification in a History field that will be appended at the bottom of the label view.

You can edit any of the fields, except the **Last Update** field. Simply click in the field cell that you want to edit. All fields, except the **Remarks** and **History** fields, contain only one line. The **Remarks** and **History** fields can contain multiple lines. Press ENTER to add a new line to these two fields.

These annotated fields are most useful when you want to search for an object stored in an EViews database. Any text that is in the fields is searchable in an EViews database; see "Querying the Database" on page 277, for further discussion.

Copying Objects

There are two distinct methods of duplicating the information in an object: copying and freezing.

If you select **Object/Copy** from the menu, EViews will create a new untitled object containing an exact copy of the original object. By exact copy, we mean that the new object duplicates all the features of the original (except for the name). It contains all of the views and procedures of the original object and can be used in future analyses just like the original object.

You may also copy an object from the workfile window. Simply highlight the object and click on **Object/Copy Selected…** or right mouse click and select **Object/Copy...**, then specify the destination name for the object.

We mention here that **Copy** is a very general and powerful operation with many additional features and uses. For example, you can copy objects across both workfiles and databases using wildcards and patterns. See "Copying Objects" on page 270, for details on these additional features.

Copy-and-Pasting Objects

The standard EViews copy command makes a copy of the object in the same workfile. When two workfiles are in memory at the same time, you may copy objects between them using copy-and-paste.

Highlight the objects you wish to copy in the source workfile. Then select **Edit/Copy** from the main menu.

Select the destination workfile by clicking on its titlebar. Then select either **Edit/Paste** or **Edit/Paste Special...** from the main menu or simply **Paste** or **Paste Special...** following a right mouse click.

Edit/Paste will perform the default paste operation. For most objects, this involves simply copying over the entire object and its contents. In other cases, the default paste operation is more involved. For example, when copy-and-pasting series between source and destination workfiles that are of different frequency, frequency conversion will be performed, if possible, using the default series settings (see "Frequency Conversion" on page 115 for additional details). EViews will place named copies of all of the highlighted objects in the destination workfile, prompting you to replace existing objects with the same name.

If you elect to **Paste Special...**, EViews will open a dialog prompting you for any relevant paste options. For example, when pasting series, you may use the dialog to override the default series settings for frequency conversion, to perform special match merging by creating links ("Series Links" on page 177). In other settings, **Paste Special...** will simply prompt you to rename the objects in the destination workfile.

Freezing Objects

The second method of copying information from an object is to *freeze* a view of the object. If you click **Object/Freeze Output** or press the **Freeze** button on the object's toolbar, a table or graph object is created that duplicates the current *view* of the original object.

Before you press **Freeze**, you are looking at a view of an object in the object window. Freezing the view makes a copy of the view and turns it into an independent object that will remain even if you delete the original object. A frozen view does not necessarily show what is currently in the original object, but rather shows a snapshot of the object at the

moment you pushed the button. For example, if you freeze a spreadsheet view of a series, you will see a view of a new *table object*; if you freeze a graphical view of a series, you will see a view of a new *graph object*.

The primary feature of freezing an object is that the tables and graphs created by freezing may be edited for presentations or reports. Frozen views do not change when the workfile sample or data change.

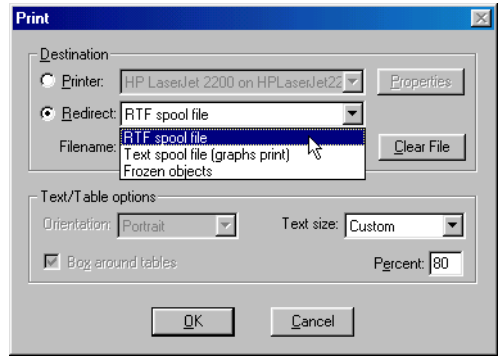
Deleting Objects

To delete an object or objects from your workfile, select the object or objects in the workfile directory. When you have selected everything you want to delete, click **Delete** or **Object/Delete Selected** on the workfile toolbar. EViews will prompt you to make certain that you wish to delete the objects.

Printing Objects

To print the currently displayed view of an object, push the **Print** button on the object window toolbar. You can also choose **File/Print** or **Object/Print** on the main EViews menu bar.

EViews will open a **Print** dialog containing the default print settings for the type of output you are printing. Here, we see the dialog for printing text information; the dialog for printing from a graph will differ slightly.



The default settings for printer type, output redirection, orientation, and text size may be set in the **Print Setup...** dialog (see "Print Setup" on page 943) or they may be overridden in the current print dialog.

For example, the print commands normally send a view or procedure output to the current Windows printer. You may specify instead that the output should be saved in the workfile as a table or graph, or spooled to an RTF or ASCII text file on disk. Simply click on **Redirect**, then select the output type from the list.

Storing Objects

EViews provides three ways to save your data on disk. You have already seen how to save entire workfiles, where all of the objects in the workfile are saved together in a single file

with the .WF1 extension. You may also store individual objects in their own *data bank* files. They may then be fetched into other workfiles.

We will defer a full discussion of storing objects to data banks and databases until Chapter 10, "EViews Databases", on page 261. For now, note that when you are working with an object, you can place it in a data bank or database file by clicking on the **Object/Store to DB...** button on the object's toolbar or menu. EViews will prompt you for additional information.

You can store several objects, by selecting them in the workfile window and then pressing the **Object/Store selected to DB...** button on the workfile toolbar or menu.

Fetching Objects

You can fetch previously stored items from a data bank or database. One of the common methods of working with data is to create a workfile and then fetch previously stored data into the workfile as needed.

To fetch objects into a workfile, select **Object/Fetch from DB...** from the workfile menu or toolbar. You will see a dialog box prompting you for additional information for the fetch: objects to be fetched, directory and database location, as applicable.

See "Fetching Objects from the Database" on page 268, for details on the advanced features of the fetch procedure.

Updating Objects

Updating works like fetching objects, but requires that the objects be present in the workfile. To update objects in the workfile, select them from the workfile window, and click on **Object/Update from DB...** from the workfile menu or toolbar. The Fetch dialog will open, but with the objects to be fetched already filled in. Simply specify the directory and database location and click **OK**.

The selected objects will be replaced by their counterparts in the data bank or database.

See Chapter 10, "EViews Databases", on page 261, for additional details on the process of updating objects from a database.

Copy-and-Paste of Object Information

You can copy the list of object information displayed in a workfile or database window to the Windows clipboard and paste the list to other program files such as word processing files or spreadsheet files. Simply highlight the objects in the workfile directory window, select **Edit/Copy** (or click anywhere in the highlighted area, with the right mouse button, and select **Copy**). Then move to the application (word processor or spreadsheet) where you want to paste the list, and select **Edit/Paste**.

If only names are displayed in the window, EViews will copy a single line containing the highlighted names to the clipboard, with each name separated by a space. If the window contains additional information, either because **View/Display Comments (Label + /–)** has been chosen in a workfile window or a query has been carried out in a database window, each name will be placed in a separate line along with the additional information.

Note that if you copy-and-paste the list of objects into another EViews workfile, the objects themselves will be copied.

Chapter 5. Basic Data Handling

The process of entering, reading, editing, manipulating, and generating data forms the foundation of most data analyses. Accordingly, most of your time in EViews will probably be spent working with data. EViews provides you with a sophisticated set of data manipulation tools that make these tasks as simple and straightforward as possible.

This chapter describes the fundamentals of working with data in EViews. There are three cornerstones of data handling in EViews: the two most common data objects, *series* and *groups*, and the use of *samples* which define the set of observations in the workfile that we wish to use in analysis.

We begin our discussion of data handling with a brief description of series, groups, and samples, and then discuss the use of these objects in basic input, output, and editing of data. Lastly, we describe the basics of frequency conversion.

In Chapter 6, "Working with Data", on page 129, we discuss the basics of EViews' powerful language for generating and manipulating the data held in series and groups. Subsequent chapters describe additional techniques and objects for working with data.

Data Objects

The actual numeric values that make up your data will generally be held in one or more of EViews' *data objects* (series, groups, matrices, vectors, and scalars). For most users, series and groups will by far be the most important objects, so they will be the primary focus of our discussion. Matrices, vectors, and scalars are discussed at greater length in the *Command and Programming Reference*.

The following discussion is intended to provide only a brief introduction to the basics of series and groups. Our goal is to describe the fundamentals of data handling in EViews. An in-depth discussion of series and group objects follows in subsequent chapters.

Series

An EViews series contains a set of observations on a numeric variable. Associated with each observation in the series is a date or observation label. For series in dated workfiles, the observations are presumed to be observed regularly over time. For undated data, the observations are not assumed to follow any particular frequency.

Note that the series object may only be used to hold numeric data. If you wish to work with alphanumeric data, you should employ alpha series. See "Alpha Series" on page 153 for discussion.

Creating a series

One method of creating a numeric series is to select **Object/New Object…** from the menu, and then to select **Series**. You may, at this time, provide a name for the series, or you can let the new series be untitled. Click **OK**. EViews will open a spreadsheet view of the new series object. All of the observations in the series will be assigned the missing value code "NA". You can then edit or use expressions to assign values for the series.

You may also use the **New Object** dialog to create alpha series. Alpha series are discussed in greater detail in "Alpha Series" on page 153.

The second method of creating a series is to generate the series using mathematical expressions. Click on **Quick/Generate Series…** in the main EViews menu, and enter an expression defining the series. We will discuss this method in depth in the next chapter.

Changing the Spreadsheet Display

EViews provides you with extensive ability to customize your series spreadsheet display.

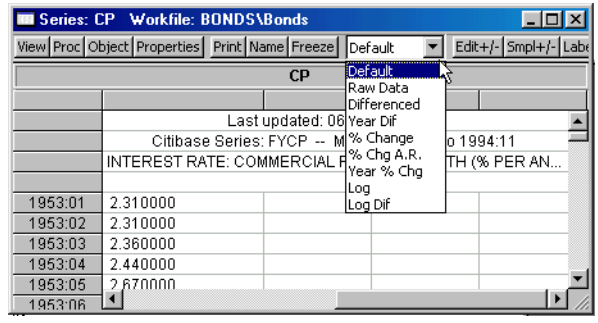
Column Widths

To resize the width of a column, simply move your mouse over the column separator and until the icon changes, then drag the column to its desired width. The new width will be remembered the next time you open the series and will be used when the series is displayed in a group spreadsheet.

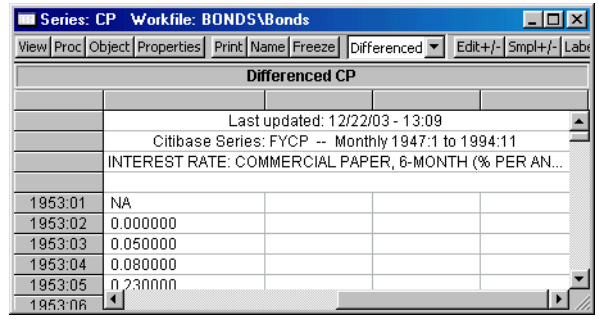
Display Type

The series display type, which is listed in the combo box in the series toolbar, determines how the series spreadsheet window shows your data.

The **Default** method shows data in either raw (underlying data) form or, if a value map is attached to the series, shows the mapped values. Alternatively, you may use the **Raw Data** to show only the underlying data. See "Value Maps" on page 163 for a description of the use of value maps.



You may also use the display type setting to show transformations of the data. You may, for example, set the display method to **Differenced**, in order to have EViews display the first-differences of your data.

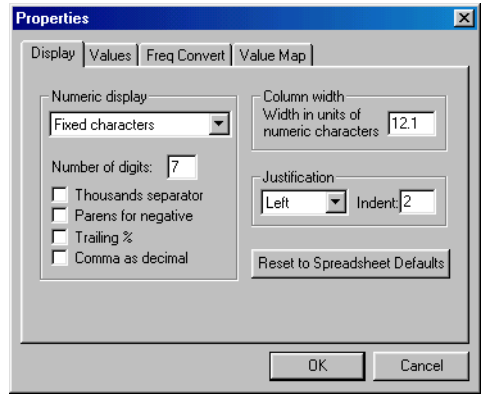

Series: CP Workfile: BONDS\Bonds
View Proc Object Properties Print Name Freeze Differenced Edit+/- Smpl+/- Labe
Differenced CP
Last updated: 12/22/03 - 13:09
Citibase Series: FYCP -- Monthly 1947:1 to 1994:11
INTEREST RATE: COMMERCIAL PAPER, 6-MONTH (% PER AN...
1953:01 NA
1953:02 0.000000
1953:03 0.050000
1953:04 0.080000
1953:05 0.230000
1953:06

Changing the display of your series values does not alter the underlying values in the series, it only modifies the values shown in the spreadsheet (the series header, located above the labels, will also change to indicate the transformation). Note, however, that if you edit the values of your series while displayed in transformed mode, EViews will change the underlying values of the series accordingly. Changing the display and editing data in transformed mode is a convenient method of inputting data that arrive as changes or other transformed values.

Display Formats

You may customize the way that numbers or characters in your series are displayed in the spreadsheet by setting the series display properties. To display the dialog, either select **View/Properties** from the series menu, click on **Properties** in the series toolbar, or right mouse click and select the **Display Format...** entry in the menu.

EViews will open the **Properties** dialog with the **Display** tab selected. You should use this dialog to change the default column width and justification for the series, and to choose from a large list of numeric display formats.



You may, for example, elect to change the display of numbers to show additional digits, to separate thousands with a comma, or to display numbers as fractions. The last four items in the **Numeric display** combo box provide options for the formatting of date number.

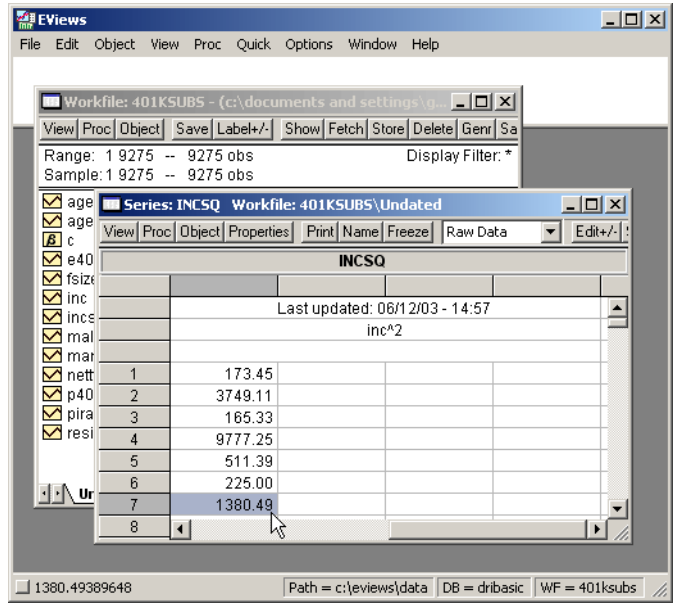
Similarly, you may elect to change the series justification by selecting **Auto**, **Left**, **Center**, or **Right**. Note that **Auto** justification will set justification to right for numeric series, and left for alpha series.

You may also use this dialog to change the column width (note that column widths in spreadsheets may also be changed interactively by dragging the column headers).

Once you click on **OK**, EViews will accept the current settings and change the spreadsheet display to reflect your choices. In addition, these display settings will be used whenever the series spreadsheet is displayed or as the default settings when the series is used in a group spreadsheet display.

Note that when you apply a display format, you may find that a portion of the contents of a cell are not visible, when, for example, the column widths are too small to show the entire cell. Alternately, you may have a numeric cell for which the current display format only shows a portion of the full precision value.

In these cases, it may be useful to examine the actual contents of a table cell. To do so, simply select the table cell. The unformatted contents of the cell will appear in the status line at the bottom of the EViews window.

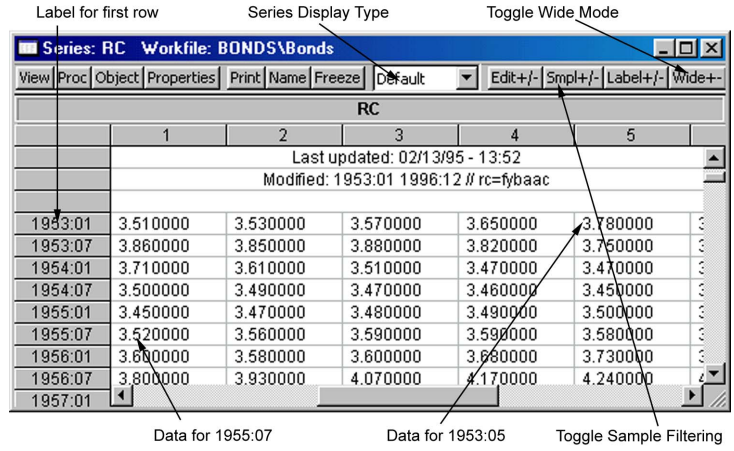


Narrow versus Wide

The *narrow display* displays the observations for the series in a single column, with date labels in the margin. The typical series spreadsheet display will use this display format.

The *wide display* arranges the observations from left to right and top to bottom, with the label for the first observation in the row displayed in the margin. For dated workfiles, EViews will, if possible, arrange the data in a form which matches the frequency of the data. Thus, semi-annual data will be displayed with two observations per row, quarterly data will contain four observations per row, and 5-day daily data will contain five observations in each row.

You can change the display to show the observations in your series in multiple columns by clicking on the **Wide** +/- button on the spreadsheet view toolbar (you may need to resize the series window to make this button visible). For example, toggling the **Wide** +/- button switches the display between the wide display (as depicted), and the narrow (single column) display.



This wide display format is useful when you wish to arrange observations for a particular season in each of the columns.

Sample Subset Display

By default, all observations in the workfile are displayed, even those observations not in the current sample. By pressing **Smpl** +/– you can toggle between showing all observations in the workfile, and showing only those observations included in the current sample.

There are two features that you should keep in mind as you toggle between the various display settings:

- If you choose to display only the observations in the current sample, EViews will switch to single column display.
- If you switch to wide display, EViews automatically turns off the display filter so that all observations in the workfile are displayed.

One consequence of this behavior is that if you begin with a narrow display of observations in the current sample, click on **Wide** +/- to switch to wide display, and then press the **Wide** +/- button again, EViews will provide a narrow display of all of the observations in the workfile. To return to the original narrow display of the current sample, you will need to press the **Smpl** +/- button again.

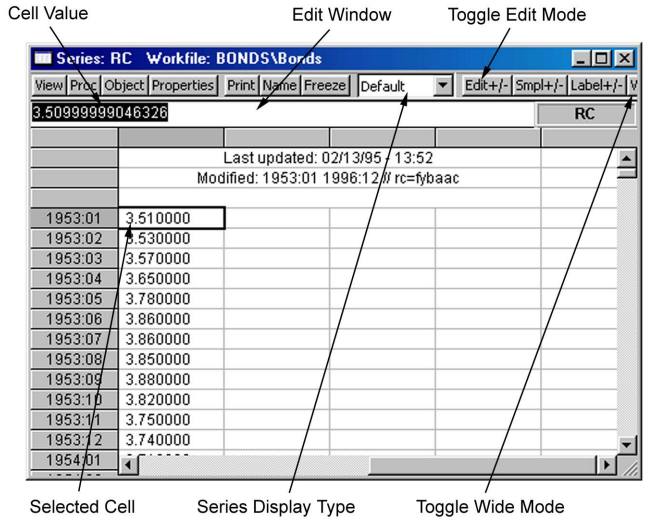
Editing a series

You can edit individual values of the data in a series.

First, open the spreadsheet view of the series. If the series window display does not show the spreadsheet view, click on the **Sheet** button, or select **View/Spreadsheet**, to change the default view.

Next, make certain that the spreadsheet window is in *edit mode*. EViews provides you with the option of protecting the data in your series by turning off the ability to edit from the spreadsheet window. You can use the **Edit +/–** button on the toolbar to toggle between *edit mode* and *protected mode*.

Here we see a series spreadsheet window in edit mode. Notice the presence of the edit window just beneath the series toolbar containing the value of RC in 1953M01, and the box around the selected cell in the spreadsheet; neither are present in protected mode.



To change the value for an observation, select the cell, type in the value, and press ENTER. For example, to change the value of RC in 1953M01, simply click on the cell containing the value, type the new value in the edit window, and press ENTER.

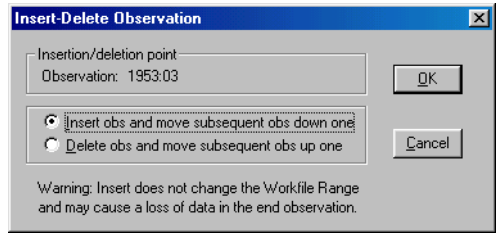
When editing series values, you should pay particular attention to the series display format, which tells you the units in which your series are displayed. Here, we see that the series values are displayed in **Default** mode so that you are editing the underlying series values (or their value mapped equivalents). Alternately, if the series were displayed in **Differenced** mode, then the edited values correspond to the first differences of the series.

Note that some cells in the spreadsheet are protected. For example, you may not edit the observation labels, or the "Last update" series label. If you select one of the protected cells, EViews will display a message in the edit window telling you that the cell cannot be edited.

When you have finished editing, you should protect yourself from inadvertently changing values of your data by clicking on **Edit +/–** to turn off edit mode.

Inserting and deleting observations in a series

You can also insert and delete observations in the series. To insert an observation, first click on the cell where you want the new observation to appear. Next, click on the **InsDel** button on the series toolbar (you may have to expand the window to make this button visible). You will see a dialog asking whether you wish to insert or delete an observation at the current position.


Insert-Delete Observation
Insertion/deletion point
Observation: 1953:03
OK
Insert obs and move subsequent obs down one
Delete obs and move subsequent obs up one
Cancel
Warning: Insert does not change the Workfile Range
and may cause a loss of data in the end observation.

If you choose to insert an observation, EViews will insert a missing value at the appropriate position and push all of the observations down so that the last observation will be lost from the workfile. If you wish to preserve this observation, you will have to expand the workfile before inserting observations. If you choose to delete an observation, all of the remaining observations will move up, so that you will have a missing value at the end of the workfile range.

Groups

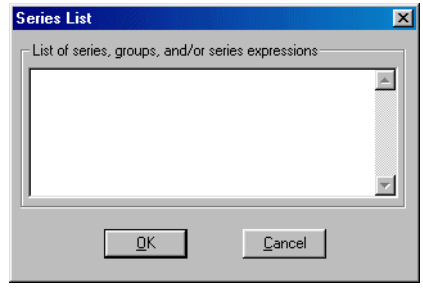
When working with multiple series, you will often want to create a group object to help you manage your data. A group is a list of series names (and potentially, mathematical expressions) that provides simultaneous access to all of the elements in the list.

With a group, you can refer to sets of variables using a single name. Thus, a set of variables may be analyzed, graphed, or printed using the group object, rather than each one of the individual series. Therefore, groups are often used in place of entering a lengthy list of names. Once a group is defined, you can use the group name in many places to refer to all of the series contained in the group.

You will also create groups of series when you wish to analyze or examine multiple series at the same time. For example, groups are used in computing correlation matrices, testing for cointegration and estimating a VAR or VEC, and graphing series against one another.

Creating Groups

There are several ways to create a group. Perhaps the easiest method is to select **Object/New Object…** from the main menu or workfile toolbar, click on **Group**, and if desired, name the object.



You should enter the names of the series to be included in the group, separated by spaces, and then click **OK**. A group window will open showing a spreadsheet view of the group.

You may have noticed that the dialog allows you to use group names and series expressions. If you include a group name, all of the series in the named group will be included in the new group. For example, suppose that the group GR1 contains the series X, Y, and Z, and you create a new group GR2, which contains GR1 and the series A and B. Then GR2 will contain X, Y, Z, A and B. Bear in mind that only the series contained in GR1, not GR1 itself, are included in GR2; if you later add series to GR1, they will not be added to GR2.

Series expressions will be discussed in greater depth later. For now, it suffices to note that series expressions are mathematical expressions that may involve one or more series (e.g. "7/2" or "3*X*Y/Z"). EViews will automatically evaluate the expressions for each observation and display the results as if they were an ordinary series. Users of spreadsheet programs will be familiar with this type of automatic recalculation.

Here, for example, is a spreadsheet view of an untitled group containing the series RC, a series expression for the lag of RG, RG(–1), and a series expression involving RC and RG.

Group: UNTITLED Workfile: BONDS\Bonds

View | Proc | Object | Print | Name | Freeze | Default | Sort | Transpose | Edit+/- | Smpl+/- | InsDe

obs	RC	RG(-1)	2*RC/RG
1953:01	3.51	NA	NA
1953:02	3.53	0.000000	NA
1953:03	3.57	0.000000	NA
1953:04	3.65	0.000000	2.579505
1953:05	3.78	2.830000	2.478689
1953:06	3.86	3.050000	2.482315
1953:07	3.86	3.110000	2.634812
1953:08	3.85	2.930000	2.610169
1953:09	3.88	2.950000	2.703833
1953:10	3.82	2.870000	2.872180
1953:11	3.75	2.660000	2.798507
1953:12	3.74	2.680000	2.888031
1954:01			

Notice here the **Default** setting for the group spreadsheet display indicates that the series RC and RG(-1) are displayed using the original values, spreadsheet types, and formats set in the original series (see "Display Formats" on page 89). A newly created group always uses the **Default** display setting, regardless of the settings in the original series, but the group does adopt the original series cell formatting. You may temporarily override the display setting by selecting a group display format. For example, to use the display settings of the original series, you should select **Series Spec**; to display differences of all of the series in the group, select **Differenced**.

An equivalent method of creating a group is to select **Quick/Show...**, or to click on the **Show** button on the workfile toolbar, and then to enter the list of series, groups and series

expressions to be included in the group. This method differs from using **Object/New Object…** only in that it does not allow you to name the object at the time it is created.

You can also create an empty group that may be used for entering new data from the keyboard or pasting data copied from another Windows program. These methods are described in detail in "Entering Data" on page 106 and "Copying-and-Pasting" on page 107.

Editing in a Group

Editing data in a group is similar to editing data in a series. Open the group window, and click on **Sheet**, if necessary, to display the spreadsheet view. If the group spreadsheet is in protected mode, click on **Edit +/–** to enable edit mode, then select a cell to edit, enter the new value, and press RETURN. The new number should appear in the spreadsheet.

Since groups are simply references to series, editing the series within a group changes the values in the original series.

As with series spreadsheet views, you may click on **Smpl +/–** to toggle between showing all of the observations in the workfile and showing only those observations in the current sample. Unlike the series window, the group window always shows series in a single column.

Note that while groups inherit many of the series display formats when they are created, to reduce confusion, groups do not initially show transformed values of the series. If you wish to edit a series in a group in transformed form, you must explicitly set a transformation type for the group display.

Samples

One of the most important concepts in EViews is the *sample* of observations. The sample is the set (often a subset) of observations in the workfile to be included in data display and in performing statistical procedures. Samples may be specified using ranges of observations and "if conditions" that observations must satisfy to be included.

For example, you can tell EViews that you want to work with observations from 1953M1 to 1970M12 and 1995M1 to 1996M12. Or you may want to work with data from 1953M1 to 1958M12 where observations in the RC series exceed 3.6.

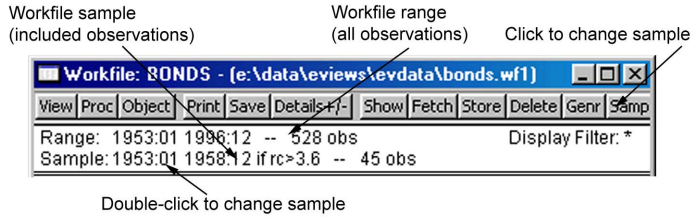
The remainder of this discussion describes the basics of using samples in non-panel workfiles. For a discussion of panel samples, see "Panel Samples" beginning on page 881.

The Workfile Sample

When you create a workfile, the *workfile sample* or *global sample* is set initially to be the entire range of the workfile. The workfile sample tells EViews what set of observations you

wish to use for subsequent operations. Unless you want to work with a different set of observations, you will not need to reset the workfile sample.

You can always determine the current workfile sample of observations by looking at the top of your workfile window.



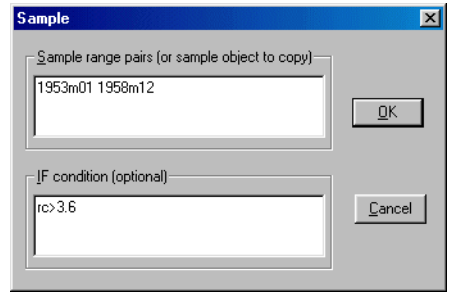
Here the BONDS workfile consists of 528 observations from January 1953 to December 1996. The current workfile sample uses a subset of those observations consisting of the 45 observations between 1953:01 and 1958:12 for which the value of the RC series exceeds 3.6.

Changing the Sample

There are four ways to set the workfile sample: you may click on the **Sample** button in the workfile toolbar, you may double click on the sample string display in the workfile window, you can select **Proc/Sample...** from the main workfile menu, or you may enter a `smpl` command in the command window. If you use one of the interactive methods, EViews will open the **Sample** dialog prompting you for input.

Date Pairs

In the upper edit field you will enter one or more pairs of dates (or observation numbers). Each pair identifies a starting and ending observation for a range to be included in the sample.



For example, if, in an annual workfile, you entered the string “1950 1980 1990 1995”, EViews will use observations for 1950 through 1980 and observations for 1990 through 1995 in subsequent operations; observations from 1981 through 1989 will be excluded. For undated data, the date pairs correspond to observation identifiers such as “1 50” for the first 50 observations.

You may enter your date pairs in a frequency other than that of the workfile. Dates used for the starts of date pairs are rounded down to the first instance of the corresponding date in the workfile frequency, while dates used for the ends of date pairs are rounded up to the last instance of the corresponding date in the workfile frequency. For example, the date pair “1990m1 2002q3” in an annual workfile will be rounded to “1990 2002”, while the

date pair "1/30/2003 7/20/2004" in a quarterly workfile will be rounded to "2003q1 2004q3".

EViews provides special keywords that may make entering sample date pairs easier. First, you can use the keyword "@ALL", to refer to the entire workfile range. In the workfile above, entering "@ALL" in the dialog is equivalent to entering "1953M1 1996M12". Furthermore, you may use "@FIRST" and "@LAST" to refer to the first and last observation in the workfile. Thus, the three sample specifications for the above workfile:

```
@all
@first 1996m12
1953m1 @last
```

are identical.

Note that when interpreting sample specifications involving days, EViews will, if necessary, use the global defaults ("Dates & Frequency Conversion" on page 939) to determine the correct ordering of days, months, and years. For example, the order of the months and years is ambiguous in the date pair:

```
1/3/91 7/5/95
```

so EViews will use the default date settings to determine the desired ordering. We caution you, however, that using the default settings to disambiguate dates in samples is not generally a good idea since a given pair may be interpreted in different ways at different times if your settings change.

Alternately, you may use the IEEE standard format, "YYYY-MM-DD", which uses a four-digit year, followed by a dash, a two-digit month, a second dash, and a two-digit day. The presence of a dash in the format means that you must enclose the date in quotes for EViews to accept this format. For example:

```
"1991-01-03" "1995-07-05"
```

will always be interpreted as January 3, 1991 and July 5, 1995. See "Free-format Conversion Details" on page 149 of the *Command and Programming Reference* for related discussion.

Sample IF conditions

The lower part of the sample dialog allows you to add conditions to the sample specification. The sample is the intersection of the set of observations defined by the range pairs in the upper window and the set of observations defined by the "if" conditions in the lower window. For example, if you enter:

Upper window: 1980 1993

Lower window: incm > 5000

the sample includes observations for 1980 through 1993 where the series INCM is greater than 5000.

Similarly, if you enter:

Upper window: 1958q1 1998q4

Lower window: gdp > gdp(-1)

all observations from the first quarter of 1958 to the last quarter of 1998, where GDP has risen from the previous quarter, will be included.

The "or" and "and" operators allow for the construction of more complex expressions. For example, suppose you now wanted to include in your analysis only those individuals whose income exceeds 5000 dollars per year and who have at least 13 years of education. Then you can enter:

Upper window: @all

Lower window: income > 5000 and educ > = 13

Multiple range pairs and "if" conditions may also be specified:

Upper window: 50 100 200 250

Lower window: income > = 4000 and educ > 12

includes undated workfile observations 50 through 100 and 200 through 250, where the series INCOME is greater than or equal to 4000 and the series EDUC is greater than 12.

You can create even more elaborate selection rules by including EViews built-in functions:

Upper window: 1958m1 1998m1

Lower window: (ed > = 6 and ed < = 13) or earn < @mean(earn)

includes all observations where the value of the variable ED falls between 6 and 13, or where the value of the variable EARN is lower than its mean. Note that you may use parentheses to group the conditions and operators when there is potential ambiguity in the order of evaluation.

It is possible that one of the comparisons used in the conditioning statement will generate a missing value. For example, if an observation on INCM is missing, then the comparison INCM > 5000 is not defined for that observation. EViews will treat such missing values as though the condition were false, and the observation will not be included in the sample.

Sample Commands

You may find it easier to set your workfile sample from the command window—instead of using the dialog, you may set the active sample using the `smpl` command. Simply click on the command window to make it active, and type the keyword "SMPL", followed by the sample string:

```
smpl 1955m1 1958m12 if rc>3.6
```

and then press ENTER (notice, in the example above, the use of the keyword "IF" to separate the two parts of the sample specification). You should see the sample change in the workfile window.

Sample Offsets

Sample range elements may contain mathematical expressions to create date offsets. This feature can be particularly useful in setting up a fixed width window of observations. For example, in the regular frequency monthly workfile above, the sample string:

```
1953m1 1953m1+11
```

defines a sample that includes the 12 observations in the calendar year beginning in 1953M1.

While EViews expects date offsets that are integer values, there is nothing to stop you from adding or subtracting non-integer values—EViews will automatically convert the number to an integer. You should be warned, however, that the conversion behavior is not guaranteed to be well-defined. If you must use non-integer values, you are strongly encouraged to use the "@ROUND", "@FLOOR" or "@CEIL" functions to enforce the desired behavior.

The offsets are perhaps most useful when combined with the special keywords to trim observations from the beginning or end of the sample. For example, to drop the first observation in your sample, you may use the sample statement:

```
smpl @first+1 @last
```

Accordingly, the following commands generate a series containing cumulative sums of the series X in XSUM:

```
smpl @first @first
series xsum = x
smpl @first+1 @last
xsum = xsum(-1) + x
```

(see "Basic Assignment" on page 138). The first two commands initialize the cumulative sum for the first observation in each cross-section. The last two commands accumulate the sum of values of X over the remaining observations.

Similarly, if you wish to estimate your equation on a subsample of data and then perform cross-validation on the last 20 observations, you may use the sample defined by,

```
smpl @first @last-20
```

to perform your estimation, and the sample,

```
smpl @last-19 @last
```

to perform your forecast evaluation.

While the use of sample offsets is generally straightforward, there are a number of important subtleties to note when working with irregular dated data and other advanced workfile structures ("Advanced Workfiles" on page 207). To understand the nuances involved, note that there are three basic steps in the handling of date offsets.

First, dates used for the starts of date pairs are rounded down to the first instance of the corresponding date in the workfile regular frequency, while dates used for the ends of date pairs are rounded up to the last instance of the corresponding date in the regular frequency. If date pairs are specified in the workfile frequency (*e.g.*, the pair "1990 2000" is used in an annual workfile), this step has no effect.

Next, EViews examines the workfile frequency date pair to determine whether the sample dates fall within the range of the observed dates in the workfile, or whether they fall outside the observed date range. The behavior of sample offsets differs in the two cases.

For simplicity of discussion, assume first that both dates fall within the range of observed dates in the workfile. In this case:

- EViews identifies base observations consisting of the earliest and latest workfile observations falling within the date pair range.
- Offsets to the date pair are then applied to the base observations by moving through the workfile observations. If, for example, the offset for the first element of a date pair is "+1", then the sample is adjusted so that it begins with the observation following the base start observation. Similarly, if the offset for the last element of a date pair is "-2", then the sample is adjusted to end two observations prior to the base end observation.

Next, we assume that both dates fall outside the range of observed workfile dates. In this setting:

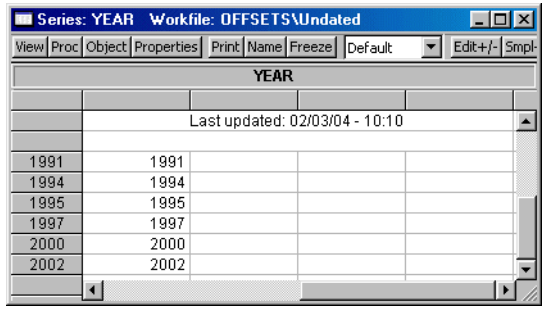
- EViews applies offsets to the date pair outside of the workfile range using the regular frequency until the earliest and latest workfile dates are reached. The base observations are then set to the earliest and latest workfile observations.
- Any remaining offsets are applied to the base observations by moving through the workfile observations, as in the earlier case.

The remaining two cases, where one element of the pair falls within, and the other element falls outside the workfile date range, follow immediately.

It is worth pointing out that the difference in behavior is not arbitrary. It follows from the fact that within the date range of the data, EViews is able to use the workfile structure to

identify an irregular calendar, but since there is no corresponding information for the dates beyond the range of the workfile, EViews is forced to use the regular frequency calendar.

A few examples will help to illustrate the basic concepts. Suppose for example, that we have an irregular dated annual workfile with observations for the years "1991", "1994", "1995", "1997", "2000", and "2002":



The sample statement:

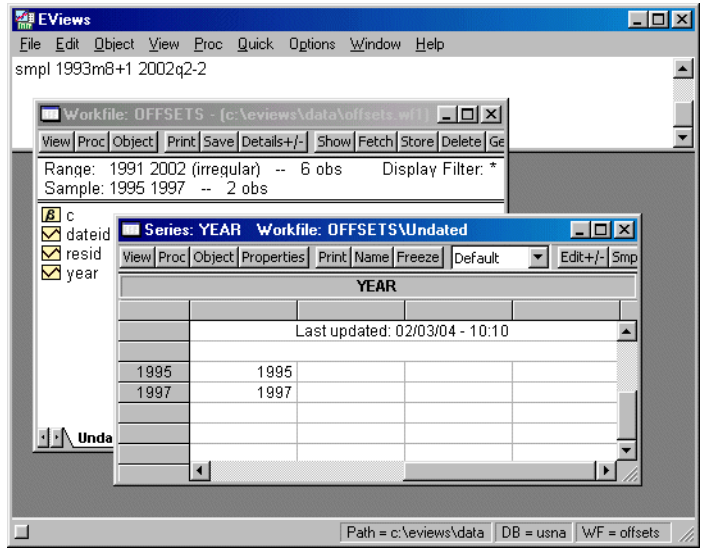
```
smpl 1993m8+1 2002q2-2
```

is processed in several steps. First, the date "1993m8" is rounded to the previous regular frequency date, "1993", and the date "2002q2" is rounded up to the last instance of the regular frequency date "2002"; thus, we have the equivalent sample statement:

```
smpl 1993+1 2002-2
```

Next, we find the base observations in the workfile corresponding to the base sample pair ("1993 2002"). The "1994" and the "2002" observations are the earliest and latest, respectively, that fall in the range.

Lastly, we apply the offsets to the remaining observations.

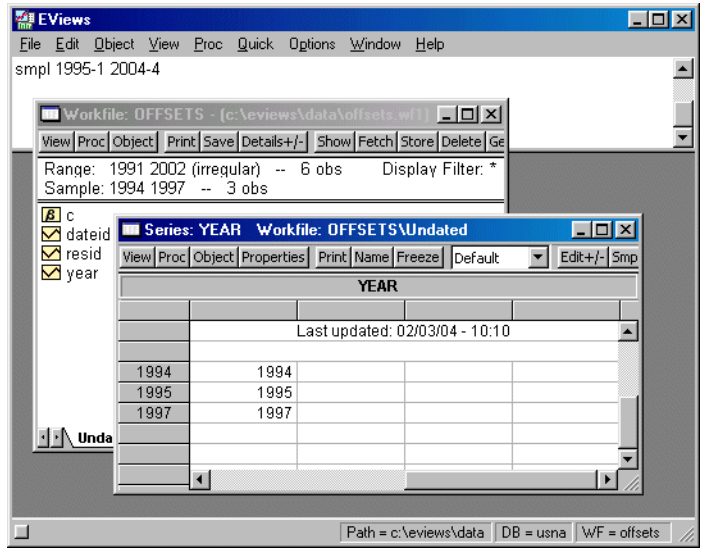

EViews
File Edit Object View Proc Quick Options Window Help
smpl 1993m8+1 2002q2-2
Workfile: OFFSETS - (c:\eviews\data\offsets.wf1)
View Proc Object Print Save Details+/- Show Fetch Store Delete Ge
Range: 1991 2002 (irregular) -- 6 obs Display Filter: *
Sample: 1995 1997 -- 2 obs
c
dateid
resid
year
Series: YEAR Workfile: OFFSETS\Undated
View Proc Object Properties Print Name Freeze Default Edit+/- Smp
YEAR
Last updated: 02/03/04 - 10:10
1995 1995
1997 1997
Unda
Path = c:\eviews\data DB = usna WF = offsets

The offsets for the start and end will drop one observation ("1994") from the beginning and two observations ("2002" and "2000") from the end of the sample, leaving two observations ("1995", "1997") in the sample.

Consider instead the sample statement:

```
smpl 1995-1 2004-4
```

In this case, no rounding is necessary since the dates are specified in the workfile frequency.


EViews
File Edit Object View Proc Quick Options Window Help
smpl 1995-1 2004-4
Workfile: OFFSETS - (c:\eviews\data\offsets.wf1)
View Proc Object Print Save Details+/- Show Fetch Store Delete Ge
Range: 1991 2002 (irregular) -- 6 obs Display Filter: *
Sample: 1994 1997 -- 3 obs
c
dateid
resid
year
Series: YEAR Workfile: OFFSETS\Undated
View Proc Object Properties Print Name Freeze Default Edit+/- Smp
YEAR
Last updated: 02/03/04 - 10:10
1994 1994
1995 1995
1997 1997
Unda
Path = c:\eviews\data DB = usna WF = offsets

For the start of the date pair, we note that the observation for "1995" corresponds to the start date. Computing the offset "-1" simply adds the "1994" observation.

For the end of the date pair, we note that "2004" is beyond the last observation in the workfile, "2002". We begin by computing offsets to "2004" using the regular fre-

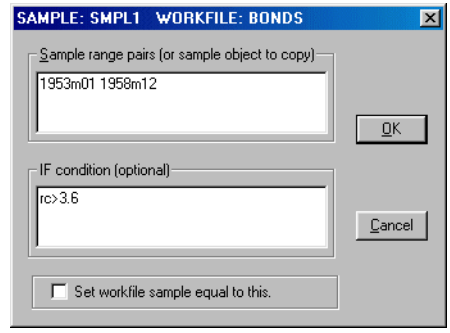
quency calendar, until we reach the highest date in the workfile, so that we "drop" the two observations "2004" and "2003". The remaining two offsets, which use the observed dates, drop the observations for "2002" and "2000". The resulting sample includes the observations "1994", "1995", and "1997".

Sample Objects

As you have seen, it is possible to develop quite elaborate selection rules for the workfile sample. However, it can become quite cumbersome and time-consuming to re-enter these rules if you change samples frequently. Fortunately, EViews provides you with a method of saving sample information in an object which can then be referred to by name. If you work with many well-defined subsets of your data, you will soon find sample objects to be indispensable.

Creating a Sample Object

To create a sample object, select **Object/New Object…** from the main menu or the workfile toolbar. When the **New Object** dialog appears, select **Sample** and, optionally provide a name. If you do not provide a name, EViews will automatically assign one for you (sample objects may not be untitled). Click on **OK** and EViews will open the sample object specification dialog:



Here is a partially filled-in sample object dialog for SMPL1. Notice that while this dialog looks very similar to the one we described above for setting the sample, there are minor cosmetic differences: the name of the sample object appears in the title bar, and there is a check box for setting the workfile sample equal to this sample object.

These cosmetic differences reflect the two distinct purposes of the dialog: (1) to define the sample object, and (2) to set the workfile sample. Since EViews separates the act of defining the sample object from the act of setting the workfile sample, you can define the object without changing the workfile sample, and vice versa.

To define the sample object, you should fill out this dialog as described before and click on **OK**. The sample object now appears in the workfile directory with a double-arrow icon.

To declare a sample object using a command, simply issue the `sample` declaration, followed by the name to be given to the sample object, and then the sample string:

```
sample mysample 1955m1 1958m12 if rc>3.6
```

EViews will create the sample object MYSAMPLE which will use observations between 1955:01 and 1958:12, where the value of the RC series is greater than 3.6.

Using a Sample Object

You may use a previously defined sample object directly to set the workfile sample. Simply open a sample object by double clicking on the name or icon. This will reopen the sample dialog. If you wish to change the sample object, you may edit the sample specification; otherwise, simply click the **Set workfile sample** check box and click on **OK**.

Or, you may set the workfile sample using the sample object, by entering the `smpl` command, followed by the sample object name. For example, the command:

```
smpl mysample
```

will set the workfile sample according to the rules contained in the sample object MYSAMPLE.

For many purposes, you may also use a named sample object as though it were an ordinary EViews series containing the values 1 and 0, for observations that are and are not included, respectively. Thus, if SMP2 is a named sample object, you may use it as though it were a series in any EViews expressions (see "Series Expressions" on page 131). For example:

```
y1*(smp2=0) + 3*y2*(smp2=1)
```

is a valid EViews expression, evaluating to the value of 3*Y2 if an observation is in SMP2, and Y1, otherwise.

You may also, for example, create a new series that is equal to a sample object, and then examine the values of the series to see which observations do and do not satisfy the sample criterion.

Additionally, one important consequence of this treatment of sample objects is that you may use sample objects in the construction of other sample objects. For example, if you create a sample object FEMALE containing observations for individuals who are females,

```
sample female @all if gender="female"
```

and a second sample object HIGHINC if INCOME is greater than 25000:

```
sample highinc @all if income>25000
```

You may set the sample to observations where individuals are low income females using:

```
smpl @all if female and not highinc
```

where we use the NOT keyword to take the complement of the observations in HIGHINC. To create a sample object HIGHFEMALE using this sample, use the command:

```
sample highfemale @all if female and not highinc
```

Alternatively, we could have used the equivalent expression

```
sample highfemale @all if female and highinc=0
```

More generally, we may use any expression involving sample objects and the keywords "AND", "OR", and "NOT", as in

```
smpl 1950 1980 if female or not highinc
```

which sets the sample to those observations from 1950 to 1980 that are also in the sample FEMALE, but not in the sample HIGHINC.

Importing Data

The data for your project may be available in a variety of forms. The data may be in a machine readable spreadsheet or text file that you created yourself or downloaded from the Internet, or perhaps they are in book or photocopy form.

There are a number of ways to read such data into EViews. Earlier, we described workfile creation tools that allow you to open data from foreign sources into a *new* workfile ("Creating a Workfile by Reading from a Foreign Data Source" on page 53). This is most likely the easiest way to get data from foreign files and database sources such as ODBC into EViews, but you should note that these tools are expressly designed for creating new workfiles.

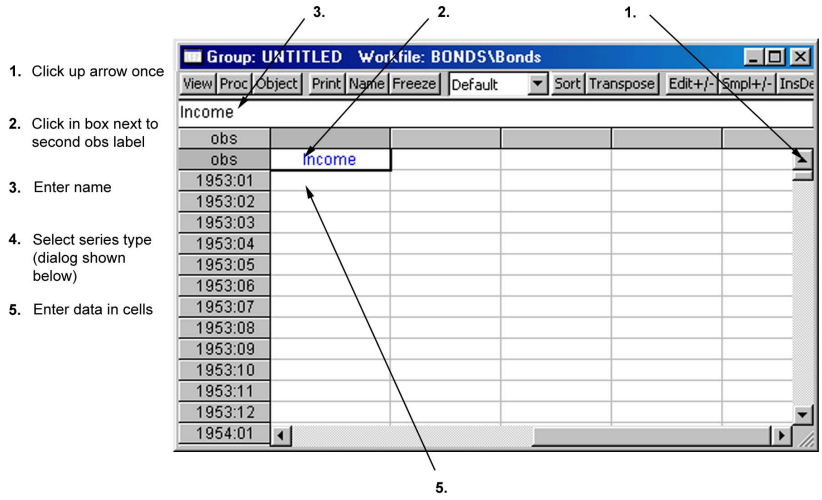
Alternatively, you may wish to import data into an *existing* workfile, perhaps into existing series in the workfile—you may, for example, wish to read a portion of an Excel file into a subset of observations in a series or group of series. We term the reading of data into existing workfiles and/or series *importing series data* to distinguish it from the creation of entirely new workfiles and series.

There are several methods for importing series data into EViews. In the remainder of this discussion, we outline the basics of data import from spreadsheet, text file, or printed formats, into series and group objects. Note that we omit, for the moment, discussion of importing data into EViews matrix, vector and pool objects, and discussion of EViews and foreign databases. The matrix and vector import tools are mentioned briefly in "Matrix Object Import" on page 112; pool import is described in "Importing Pooled Data" on page 832; EViews databases are the subject of Chapter 10, "EViews Databases", beginning on page 261.

Entering Data

For small datasets in printed form, you may wish to enter the data by typing at the keyboard.

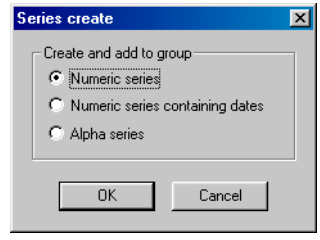
- Your first step is to open a temporary spreadsheet window in which you will enter the data. Choose **Quick/Empty Group (Edit Series)** from the main menu to open an untitled group window:



- The next step is to create and name the series. First click once on the up arrow in the scroll bar to display the second **obs** label on the left-hand column. The row of cells next to the second **obs** label is where you will enter and edit series names.

  Click once in the cell next to the second **obs** label, and enter your first series name. Here we have typed "income" in the edit window (the name in the cell changes as we type in the edit window). Press RETURN. If you enter the name of an existing series, the series data will be brought into the group.

- EViews will prompt you to specify a series type for the column. You may select a numeric series, numeric series containing date values, or an alpha series. When you click on **OK**, EViews will create a numeric or alpha series and will apply formatting information that will aid you in viewing your data.



- You should repeat this procedure in subsequent columns for each additional series. If you decide you want to rename one of your series, simply select the cell containing the series name,

edit the name in the edit window, and then press RETURN. EViews will prompt you to confirm the series rename.

- To enter the data, click on the appropriate cell and type the number or text. Pressing RETURN after entering the cell value will move you to the next cell. If you prefer, you can use the cursor keys to navigate the spreadsheet.
- When you are finished entering data, close the group window. If you wish, you can first name the untitled group by clicking on the **Name** button. Otherwise, if you do not wish to keep the group, answer **Yes** when EViews asks you to confirm the deletion.

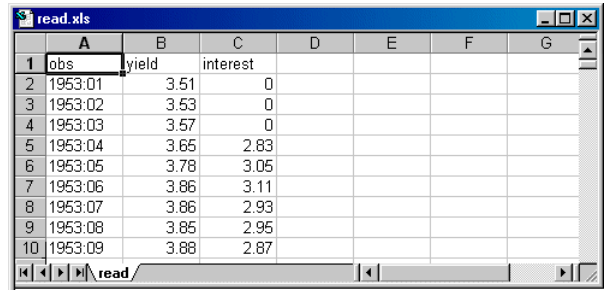
Copying-and-Pasting

The Windows clipboard is a handy way to move data within EViews and between EViews and other software applications. It is a natural tool for importing data from Excel and other Windows applications that support Windows copy-and-paste.

Copying from Windows applications

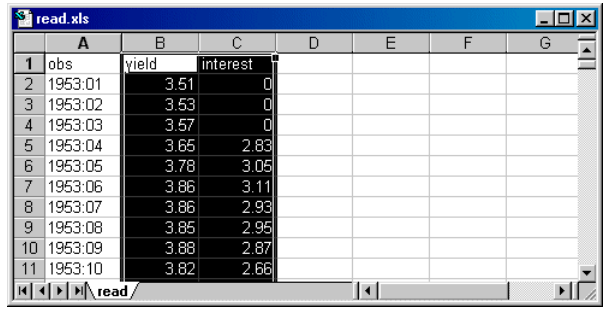
The following discussion involves an example using an Excel spreadsheet, but the basic principles apply for other Windows applications.

Suppose you have bond yield and interest rate data in an Excel spreadsheet that you would like to bring into EViews.



Open the spreadsheet in Excel. Your first step is to highlight the cells to be imported into EViews. Since the column headings YIELD and INTEREST will be used as EViews variable names, you should highlight them as well. Since EViews understands dated data, and we are going to create a monthly workfile, you do not need to copy the date column. Instead, click on the column label B and drag to the column label C. The two columns of the spreadsheet will be highlighted.

Select **Edit/Copy** to copy the highlighted data to the clipboard.



	A	B	C	D	E	F	G
1	obs	yield	interest				
2	1953:01	3.51	0				
3	1953:02	3.53	0				
4	1953:03	3.57	0				
5	1953:04	3.65	2.83				
6	1953:05	3.78	3.05				
7	1953:06	3.86	3.11				
8	1953:07	3.86	2.93				
9	1953:08	3.85	2.95				
10	1953:09	3.88	2.87				
11	1953:10	3.82	2.66				

Pasting into New Series

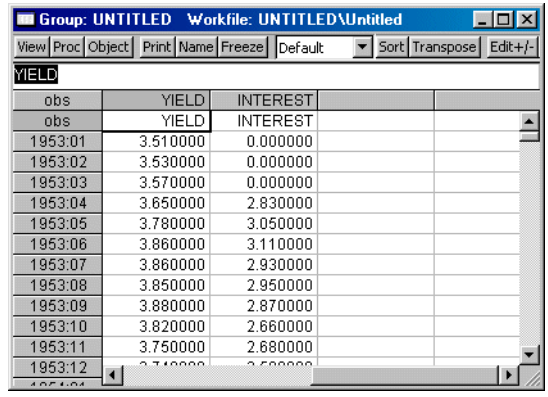
Start EViews and create a new, or load an existing, monthly workfile containing the dates in the Excel spreadsheet (in our example, 1953:1 through 1994:11). Make certain that the sample is set to include the same observations that you have copied onto the clipboard.

Select **Quick/Empty Group (Edit Series)**. Note that the spreadsheet opens in edit mode so there is no need to click the **Edit +/–** button.

Here, we have created a monthly workfile with a range from 1953:1 to 1999:12. The first row of the EViews spreadsheet is labeled 1953:01. Since we are pasting in the series names as well, you should click on the up arrow in the scroll bar to make room for the series names.

Place the cursor in the upper-left cell, just to the right of the second **obs** label. Then select **Edit/Paste** from the main menu (*not* **Edit +/–** in the toolbar). The group spreadsheet will now contain the data from the clipboard.

EViews automatically analyzes the data on the clipboard to determine the most likely series type. If, for example, your series contains text that can always be interpreted as a number, EViews will create a numeric series. Here, the numeric series YIELD and INTEREST have been created in the workfile.



obs	YIELD	INTEREST
obs	YIELD	INTEREST
1953:01	3.510000	0.000000
1953:02	3.530000	0.000000
1953:03	3.570000	0.000000
1953:04	3.650000	2.830000
1953:05	3.780000	3.050000
1953:06	3.860000	3.110000
1953:07	3.860000	2.930000
1953:08	3.850000	2.950000
1953:09	3.880000	2.870000
1953:10	3.820000	2.660000
1953:11	3.750000	2.680000
1953:12		

If the numbers in the series may all be interpreted as date values, or if the data are all string representations of dates, EViews will create a numeric series formatted to display dates.

If you paste a name corresponding to an object that already exists in the workfile, EViews will find the next available name by appending an integer to the series name. For example, if SER already exists in the workfile, pasting the name “SER” will create a series SER01.

You may now close the group window and delete the untitled group without losing the two series.

Pasting into Existing Series

You can import data from the clipboard into an existing EViews series or group spreadsheet by using **Edit/Paste** in the same fashion. There are only a few additional issues to consider:

- To paste several series, you will first open a group window containing the existing series. The easiest way to do this is to click on **Show**, and then type the series names in the order they appear on the clipboard. Alternatively, you can create an untitled group by selecting the first series, holding down the **Ctrl**-key and click select each subsequent series (in order), and then double clicking to open.
- Make certain that the group window is showing the sample range that corresponds to the data on the clipboard.
- Next, make certain that the group window is in edit mode. If not in edit mode, press the **Edit +/–** button to toggle to edit mode. Place the cursor in the target cell, and select **Edit/Paste** from the main menu.
- Finally, click on **Edit +/–** to return to protected mode.
- If you are pasting into a single series you will need to make certain that the series window is in edit mode, and that the series is viewed in a single column. If the series is in multiple columns, push on the **Wide +/–** button. Then **Edit/Paste** the data as usual, and click on **Edit +/–** to protect the data.

Importing Data from a Spreadsheet or Text File

You can also read data directly from files created by other programs. Data may be in standard ASCII form or in either Lotus (.WKS, .WK1 or .WK3) or Excel (.XLS) spreadsheet formats.

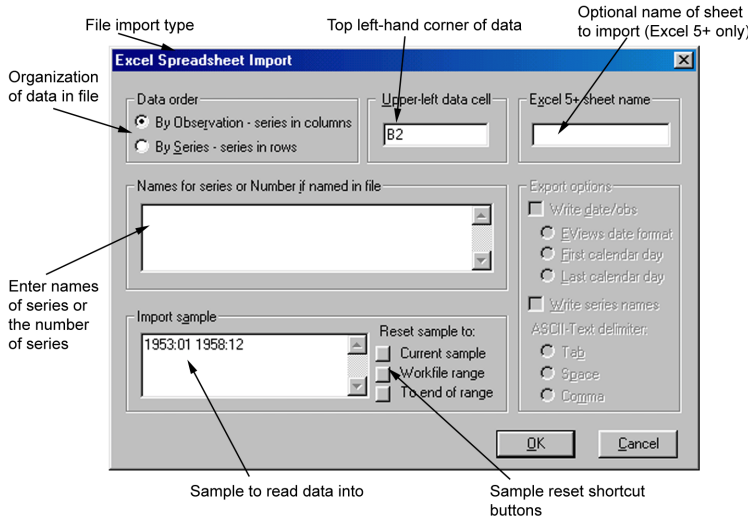
First, make certain that you have an open workfile to receive the contents of the data import and that the workfile window is active.

Next, click on **Proc/Import/Read Text-Lotus-Excel...** You will see a standard **File Open** dialog box asking you to specify the type and name of the file. Select a file type, navigate to the directory containing the file, and double click on the name. Alternatively, type in the name of the file that you wish to read (with full path information, if appropriate); if possible, EViews will automatically set the file type, otherwise it will treat the file as an ASCII file. Click on **Open**.

EViews will open a dialog prompting you for additional information about the import procedure. The dialog will differ greatly depending on whether the source file is a spreadsheet or an ASCII file.

Spreadsheet Import

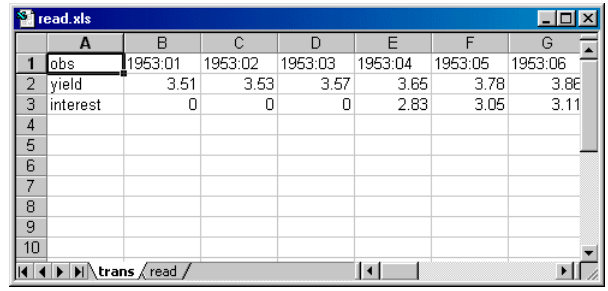
The title bar of the dialog will identify the type of file that you have asked EViews to read. Here is the dialog for importing an Excel 5 (or later versions of Excel) spreadsheet:



You will see slightly different versions of this dialog depending on whether you are reading a Lotus or an Excel 4 (and earlier) file. Now fill in the dialog:

- First, you need to tell EViews whether the data are ordered by observation or by series. **By observation** means that all of the data for the first observation are followed by all of the data for the second observation, etc. **By series** means that all of the data for the first variable are followed by all data for the second variable, etc. Another interpretation for "by observation" is that variables are arranged in columns while "by series" implies that all of the observations for a variable are in a single row.

Our Excel example above ("Copying from Windows applications" on page 107) is organized by observation since each series is in a separate column. If the Excel data for YIELD and INTERREST were each contained in a single row as depicted here, then the data should be read by series.



- Next, tell EViews the location of the beginning cell (upper left-hand corner) of your actual data, not including any label or date information. In both examples above, the upper left-hand cell is B2.

- In the edit box in the middle of the dialog, enter the names to be assigned to the series you will be importing. EViews reads spreadsheet data in contiguous blocks, so you should provide a name for each column or row (depending on the orientation of the data), even if you only wish to read selected columns or rows. To read a column or row into an alpha series, you should enter the tag "$" following the series name (*e.g.*, "NAME $ INCOME CONSUMP").

- Alternatively, if the names that you wish to use for your series are contained in the file, you can simply provide the number of series to be read. The names must be adjacent to your data. If the data are organized by row and the starting cell is B2, then the names must be in column A, beginning at cell A2. If the data are organized by column beginning in B2, then the names must be in row 1, starting in cell B1. If, in the course of reading the data, EViews encounters an invalid cell name, it will automatically assign the next unused name with the prefix SER, followed by a number (*e.g.*, SER01, SER02, *etc.*).

- Lastly, you should tell EViews the sample of data that you wish to import. EViews begins with the first observation in the file and assigns it to the first date in the sample for each variable. Each successive observation in the file is associated with successive observations in the sample. Thus, in an annual workfile, if you enter the sample:

  ```
  1971 1975 1990 1991
  ```

  in the import dialog, the first five observations will be assigned to the dates 1971–1975, and the sixth and seventh observations will be assigned to the dates 1990–1991. The data in the intermediate period will be unaffected by the importing procedure.

You should be warned that if you read into a sample which has more observations than are present in your input file, observations for which there are no corresponding inputs will be assigned missing values. For example, if you read into the sample defined as "1971 1990", and there are only 10 observations in the input file, the observations from 1981 to 1990 will be assigned missing values.

When the dialog is first displayed, EViews enters the current workfile sample in the edit box by default. You should edit this string to reflect the desired sample. To make it easier to set the sample, EViews provides you with three push-buttons which change the string in the edit box to commonly used values:

1. **Current sample** sets the dialog string to the current workfile sample.
2. **Workfile range** sets the dialog string to the entire range of the workfile.
3. **To end of range** sets the dialog string to all observations from the beginning of the current sample to the end of the workfile range.

- If you are reading data from an Excel 5 workbook file, there will be an additional edit box where you can enter the name of the sheet containing your data. If you do not enter a name, EViews will read from the topmost sheet in the Excel workbook.
- When the dialog is completely filled out, simply click **OK** and EViews will read your file, creating series and assigning values as requested.

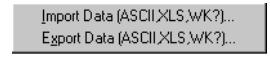
ASCII Import

If you choose to read from an ASCII file, EViews will open an ASCII Text Import dialog. Fill out the dialog to read from the specified file.

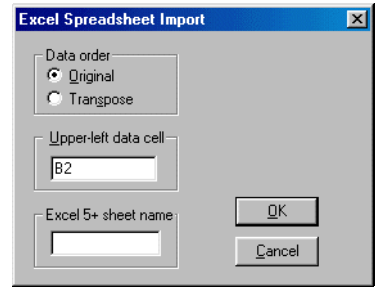
The dialog box for ASCII file import is considerably more complicated than the corresponding spreadsheet dialog. While unfortunate, this complexity is necessary since there is no standard format for ASCII files. EViews provides you with a range of options to handle various types of ASCII files.

ASCII file importing is explained in considerable detail in "Importing ASCII Text Files" beginning on page 120.

Matrix Object Import



The preceding discussion focused on importing data into series or group objects. Similar tools are available for importing data directly into a matrix object from spreadsheet or from ASCII text files.



To import data from a file into a matrix object, you must first open the correctly sized matrix object, and select **Proc/Import Data (ASCII, .XLS, .WK?)...**. After you select your file, EViews will open an import dialog.

Here, we depict the dialog for importing from an Excel spreadsheet. The corresponding ASCII dialog has many more options, since ASCII file reading is more complicated. Note that both the import and export dialogs differ little from the series import dialogs described above. The differences reflect the different nature of series and matrix input and output. For example, dialog options for series names and the sample are omitted since they do not apply to matrices.

In reading from a file, EViews first fills the matrix with NAs, puts the first data element in the (1,1) element of the matrix, and then continues reading the data by row or column according to the specified settings for **Data order**. If this option is set as **Original**, EViews will read by row, filling the first row from left to right, and then continuing on to the next row. If the ordering is set as **Transpose**, EViews will read by column, reading the first column from top to bottom and then continuing on to the next column. In either case, the data read from the file are placed into the matrix by row.

ASCII files provide you with the option of reading your file as a rectangle. If your ASCII file is laid out as a rectangle, the contents of the rectangle will be placed in the matrix beginning at the (1,1) element of the matrix. For example, if you have a 3×3 matrix X in EViews, and read from the ASCII file containing:

```
1 2 3 4
5 6 7 8
9 10 11 12
```

using the **File laid out as rectangle** option, the matrix X will contain the corresponding rectangular portion of the ASCII file:

```
1 2 3
5 6 7
9 10 11
```

If you do not select the rectangular read option, EViews fills the matrix element-by-element, reading from the file line-by-line Then X will contain:

```
1 2 3
4 5 6
7 8 9
```

Exporting Data

EViews provides you with a number of methods for getting data from EViews into other applications.

Copying and Pasting

You can click and drag in a spreadsheet view or table of statistical results to highlight the cells you want to copy. Then click **Edit/Copy…** in the main menu to put the data into the clipboard. You will see a dialog box asking whether to copy the numbers with the precision showing on your screen (formatted copy) or to copy the numbers at full precision (unformatted copy).

As a shortcut, you can highlight entire rows or columns of cells by clicking on the gray border that surrounds the spreadsheet. Dragging across the border selects multiple rows or columns. To copy several adjacent series from the spreadsheet, drag across their names in the top border. All of their data will be highlighted. Then click **Edit/Copy…** to put the data into the clipboard.

Once the data are on the clipboard, switch to the target application, and select **Edit/Paste**.

Exporting to a Spreadsheet or Text File

First, click on **Proc/Export/Write Text-Lotus-Excel…** from the workfile toolbar or main menu, then enter the name and type of the output file in the **SaveAs** dialog. As you fill out the **SaveAs** dialog, keep in mind the following behavior:

- If you enter a file name with an extension, EViews will use the file extension to identify the file type. Files with common spreadsheet extensions (".XLS", ".WK3", ".WK1", and ".WKS") will be saved to the appropriate spreadsheet type. All others will be saved as ASCII files.
- If you do not enter an extension, EViews will use the file type selected in the combobox to determine the output type. Spreadsheet files will have the appropriate extensions appended to the name. ASCII files will be saved using the name provided in the dialog, without an extension. EViews will not append extensions to ASCII files unless you explicitly include one in the file name.
- Note that EViews cannot, at present, write into an existing file. The file that you select will, if necessary, be replaced.

Once you have specified the output file, click **OK** to open the export dialog.

Tip: if you highlight the series you wish to export before beginning the export procedure, the series names will be used to fill out the export dialog.

Spreadsheet Export

The dialogs for spreadsheet export are virtually identical to the dialogs for spreadsheet import. You should determine the orientation of your data, the series to export, and the sample of observations to be written.

Additionally, EViews provides you with checkboxes for determining whether to include the series names and/or the series dates in the spreadsheet. If you choose to write one or both to the spreadsheet, make certain that the starting cell for your data leaves the necessary room along the borders for the information. If the necessary room is not available, EViews will ignore the option—for example, if you choose to write your data beginning in cell A1, EViews will not write the names or dates.

ASCII Export

The ASCII export dialog is quite similar to the spreadsheet export dialog, but it contains a few additional options:

- You can change the text string to be used for writing missing values. Simply enter the text string in the edit field.
- EViews provides you with the option of separating data values with a tab, a space, or a comma. Click on the desired radio button.

We caution that if you attempt to write your data by series, EViews will write all of the observations for a series on a single line. If you have a reasonably long series of observations, these data may overflow the line-length of other programs.

Matrix Object Export

Exporting data from a matrix object simply reverses the matrix import ("Matrix Object Import" on page 112). To write the contents of the matrix to a file, select **Proc/Export Data (ASCII, .XLS, .WK?)...** from the matrix toolbar and fill in the dialog as appropriate.

Frequency Conversion

Every series in EViews has an associated frequency. When a series is in a workfile, the series is stored at the frequency of the workfile. When a series is held in a database (Chapter 10, "EViews Databases"), it is stored at its own frequency. Since all series in the same workfile page must share a common frequency, moving a series from one workfile to another or from a database to a workfile page will cause the series being moved to be converted to the frequency of the workfile page into which it is being placed.

Performing Frequency Conversion

Frequency conversion is performed in EViews simply by copying or fetching a series with one frequency into a workfile of another frequency.

Copy-and-Paste

Suppose that you have two workfile page (or a source database and a destination workfile page), where the source contains quarterly data on the series YQ, and the destination workfile contains annual data. Note that you may copy between pages in the same workfile or between separate workfiles.

To convert YQ from a quarterly to annual frequency, you may copy-and-paste the series from the source quarterly workfile to the annual workfile. Click on the YQ series in the quarterly workfile, press the right-mouse button and select **Copy**, navigate to the annual workfile, then right mouse button and select **Paste** or **Paste Special...**.

If you select **Paste**, EViews will copy YQ to the annual page, using the default frequency conversion settings present in YQ to perform the conversion.

If you select **Paste Special...**, EViews will display a dialog offering you the opportunity to override the default frequency conversion settings. Before describing this dialog ("Overriding Default Conversion Methods" on page 120), we provide a background on frequency conversion methods, and describe how default conversion methods are specified in EViews.

Using Commands

You may use either the `copy` or `fetch` command to move series between workfiles or between a database and a workfile. EViews will perform frequency conversion if the frequencies of the source and destination do not match.

See `copy` (p. 249) and `fetch` (p. 291) in the *Command and Programming Reference* for details.

Frequency Conversion Methods

There are three types of frequency conversion: high frequency to low frequency conversion, low frequency to high frequency conversion, and frequency conversion between a dated and undated workfile.

EViews provides you with the ability to specify methods for all types of conversion. In addition, there are settings that control the handling of missing values when performing the conversion.

High Frequency to Low Frequency

If a numeric series being imported has a higher frequency than the workfile, you may choose between a number of different conversion methods:

- Average observations
- Sum observations

- First observation
- Last observation
- Maximum observation
- Minimum observation
- No down conversions

with the latter setting permitting you to disallow high to low conversions. In this case, EViews will generate an error if you attempt to convert from high to low frequency.

In addition, you may specify how EViews handles missing data when carrying out the calculations. You may elect to propagate NAs so that whenever a missing value appears in a calculation, the result for the corresponding period will be an NA. Alternatively, you may elect not to propagate NAs so that calculations will be performed ignoring the missing values (though if all values for a period are missing, the corresponding result will still be an NA).

Low Frequency to High Frequency

EViews also provides a number of different interpolation methods for dealing with the case where the series being brought into the workfile has a lower frequency than the workfile. Since observing a series at a lower frequency provides fundamentally less information than observing the same series at a higher frequency, it is generally not possible to recover the high frequency series from the low frequency data. Consequently, the results from EViews' interpolation methods should be considered to be suggestive rather than providing the true values of the underlying series.

EViews supports the following interpolation methods:

- **Constant**: Constant with sum or average matched to the source data.
- **Quadratic**: Local quadratic with sum or average matched to the source data.
- **Linear**: Linear with last observation matched to the source data.
- **Cubic**: Cubic spline with last observation matched to the source data.
- **No conversion**: Do not allow up conversion.

Using an interpolation method which matches the average means that the average of the interpolated points for each period is equal to the source data point for that period. Similarly if the sum is matched, the interpolated points will sum to the source data point for the period, and if the last observation is matched, the last interpolated point will equal the source data point for the period.

For all methods, all relevant data from the low frequency series is used when forming the high frequency series, even if the destination observations are a subset of the observations available in the source.

The following describes the different methods in greater detail:

- **Constant: match average, Constant: match sum**—These two methods assign the same value to all observations in the high frequency series associated with a particular low frequency period. In one case, the value is chosen so that the average of the high frequency observation matches the low frequency observation (the value is simply repeated). In the other case, the value is chosen so that the sum of the high frequency observations matches the low frequency observation (the value is divided by the number of observations).
- **Quadratic: match average, Quadratic: match sum**—These two methods fit a local quadratic polynomial for each observation of the low frequency series, then use this polynomial to fill in all observations of the high frequency series associated with the period. The quadratic polynomial is formed by taking sets of three adjacent points from the source series and fitting a quadratic so that either the average or the sum of the high frequency points matches the low frequency data actually observed. For most points, one point before and one point after the period currently being interpolated are used to provide the three points. For end points, the two periods are both taken from the one side where data is available.

  This method is a purely local method. The resulting interpolation curves are not constrained to be continuous at the boundaries between adjacent periods. Because of this, the method is better suited to situations where relatively few data points are being interpolated and the source data is fairly smooth.
- **Linear: match last**—This method assigns each value in the low frequency series to the last high frequency observation associated with the low frequency period, then places all intermediate points on straight lines connecting these points.
- **Cubic: match last**—This method assigns each value in the low frequency series to the last high frequency observation associated with the low frequency period, then places all intermediate points on a natural cubic spline connecting all the points.

  A natural cubic spline is defined by the following properties:

  1. Each segment of the curve is represented by a cubic polynomial.
  2. Adjacent segments of the curve have the same level, first derivative and second derivative at the point where they meet.
  3. The second derivative of the curve at the two global end points is equal to zero (this is the “natural” spline condition).

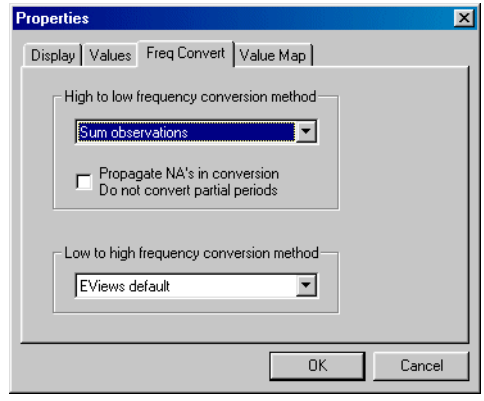
Cubic spline interpolation is a global interpolation method so that changing any one point (or adding an additional point) to the source series will affect all points in the interpolated series.

Undated Conversion

If you fetch or copy a series to or from an undated or unstructured workfile into or from a dated workfile, the data will be copied sequentially, beginning at the starting observation number of the undated or unstructured series (generally the first observation).

Specifying Default Conversion Methods

When performing frequency conversion of one or more series, EViews uses the default settings in each series to perform the conversion. These settings may be specified in each series using the **Freq Convert** tab of the **Properties** dialog. To access the dialog, select **View/Properties...** from the series main menu, or click on the **Properties** button on the series toolbar.



If the series default setting is set to **EViews default**, the series will take its frequency conversion setting from the EViews global options (see "Dates & Frequency Conversion" on page 939 in Appendix A, "Global Options"). Here, the high to low conversion is set to **Sum observations**, overriding the global setting, while the low to high uses the **EViews default** global setting.

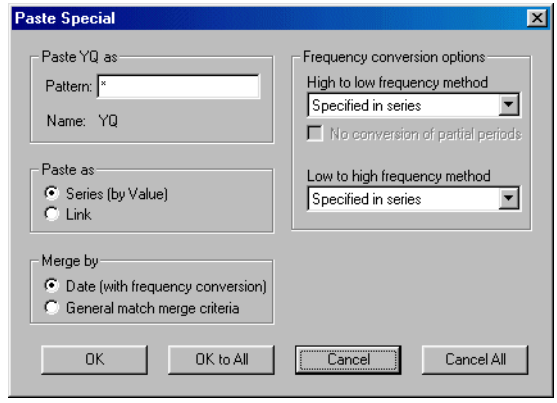
This two level default system allows you to set global default settings for frequency conversion that apply to all newly created series, while allowing you to override the default settings for specific series.

As an example of controlling frequency conversion using default settings, suppose you have daily data consisting of HIGH, LOW, and CLOSE series for a particular stock, from which you would like to construct a monthly workfile. If you use the default frequency conversion methods, the monthly workfile will contain series which use the series defaults, which is not likely to be what you want. By setting the frequency conversion method of the HIGH series to **Max observation**, of the LOW series to **Min observation**, and of the CLOSE series to **Last observation**, you may use conversion to populate a monthly workfile with converted daily data that follow the desired behavior.

Overriding Default Conversion Methods

If you use copy-and-paste to copy one or more series between two workfiles, EViews will copy the series to the destination page, using the default frequency conversion settings present in the series to perform the conversion.

If, when pasting the series into the destination, you use **Paste Special...** in place of **Paste**, EViews will display a dialog offering you the opportunity to override the default frequency conversion settings.



You need not concern yourself with most of the settings in this dialog at the moment; the dialog is discussed in greater detail in "Frequency conversion links" on page 198.

We note, however, that the dialog offers us the opportunity to change both the name of the pasted YQ series, and the frequency conversion method.

The "*" wildcard in the Pattern field is used to indicate that we will use the original name (wildcards are most useful when pasting multiple series). We may edit the field to provide a name or alternate wildcard pattern. For example, changing this setting to "*A" would copy the YQ series as YQA in the destination workfile.

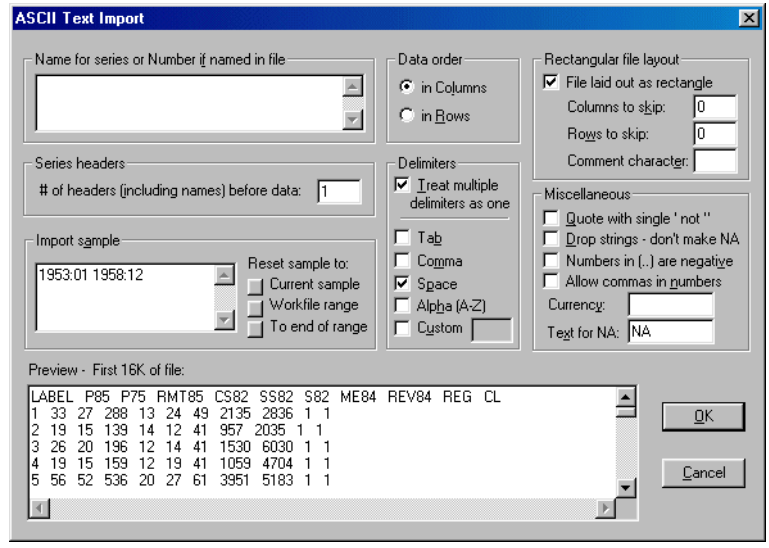
Additionally, we note that the dialog allows us to use the frequency conversion method **Specified in series** or to select alternative methods.

If, instead of copy-and-paste, you are using either the `copy` or `fetch` command and you provide an option to set the conversion method, then EViews will use this method for all of the series listed in the command (see `copy` (p. 249) and `fetch` (p. 291) in the *Command and Programming Reference* for details).

Importing ASCII Text Files

To import an ASCII text file, click on **Proc/Import/Read Text-Lotus-Excel...** from the main menu or the workfile toolbar, and select the file in the File Open dialog. The **ASCII Text Import** dialog will be displayed.

You may notice that the dialog is more complicated than the corresponding spreadsheet dialog. Since there is no standard format for ASCII text files, we need to provide a variety of options to handle various types of files.


ASCII Text Import
Name for series or Number if named in file
Data order
in Columns
in Rows
Rectangular file layout
File laid out as rectangle
Columns to skip:
Rows to skip:
Comment character:
Series headers
# of headers (including names) before data:
Delimiters
Treat multiple delimiters as one
Tab
Comma
Space
Alpha (A-Z)
Custom
Miscellaneous
Quote with single ' not ''
Drop strings - don't make NA
Numbers in (..) are negative
Allow commas in numbers
Currency:
Text for NA:
NA
Import sample
1953:01 1958:12
Reset sample to:
Current sample
Workfile range
To end of range
Preview - First 16K of file:
LABEL P85 P75 RMT85 CS82 SS82 S82 ME84 REV84 REG CL
1 33 27 288 13 24 49 2135 2836 1 1
2 19 15 139 14 12 41 957 2035 1 1
3 26 20 196 12 14 41 1530 6030 1 1
4 19 15 159 12 19 41 1059 4704 1 1
5 56 52 536 20 27 61 3951 5183 1 1
OK
Cancel

Note that the preview window at the bottom of the dialog shows you the first 16K of your file. You can use this information to set the various formatting options in the dialog.

You must provide the following information:

- **Names for series or Number of series if names in file**. If the file does not contain series names, or if you do not want to use the names in the file, list the names of the series in the order they appear in the file, separated by spaces.

  If the names of the series are located in the file *before the start of the data*, you can tell EViews to use these names by entering a number representing the number of series to be read.

  If possible, you should avoid using parentheses and mathematical symbols such as "*", "+", "–", "/", "^" in the series names in the file. If EViews tries to read the names from the file and encounters an invalid name, it will try to rename the series to a valid name by replacing invalid characters with underscores and numbers. For example, if the series is named "X(-3)" in the file, EViews will rename this series to "X__3_01". If "X__3_01" is already a series name, then EViews will name the series "X__3_02", and so forth.

  If EViews cannot name your series, say, because the name is a reserved name, or because the name is used by an object that is not a series, the series will be named "SER01", "SER02", *etc.*

You should be very careful in naming your series and listing the names in the dialog. If the name in the list or in the file is the same as an existing series name in the workfile, the *data in the existing series will be overwritten*.

- **Data order**. You need to specify how the data are organized in your file. If your data are ordered by observation so that each series is in a column, select **in Columns**. If your data are ordered by series so that all the data for the first series are in one row followed by all the data for the second series, and so on, select, **in Rows**.

- **Import sample**. You should specify the sample in which to place the data from the file. EViews fills out the dialog with the current workfile sample, but you can edit the sample string or use the sample reset buttons to change the input sample. The input sample only sets the sample for the import procedure, it does not alter the workfile sample.

  EViews fills all of the observations in the current sample using the data in the input file. There are a couple of rules to keep in mind:

  1. EViews assigns values to all observations in the input sample. Observations outside of the input sample will not be changed.
  2. If there are too few values in the input file, EViews will assign NAs to the extra observations in the sample.
  3. Once all of the data for the sample have been read, the remainder of the input file will be ignored.

In addition to the above information, you can use the following options to further control the way EViews reads in ASCII data.

EViews scans the first few lines of the source file and sets the default formatting options in the dialog based on what it finds. However, these settings are based on a limited number of lines and may not be appropriate. You may find that you need to reset these options.

Delimiters

Delimiters are the characters that your file uses to separate observations. You can specify multiple delimiters by selecting the appropriate entries. **Tab**, **Comma**, and **Space** are self-explanatory. The **Alpha** option treats any of the 26 characters from the alphabet as a delimiter.

For delimiters not listed in the option list, you can select the **Custom** option and specify the symbols you wish to treat as delimiters. For example, you can treat the slash "/" as a delimiter by selecting **Custom** and entering the character in the edit box. If you enter more than one character, *each* character will be treated as a delimiter. For example, if you enter double slash "//" in the **Custom** field, then the single slash "/" will be treated as a delim-

iter, instead of the double slash "//". The double slash will be interpreted as two delimiters.

EViews provides you with the option of treating multiple delimiter characters as a single delimiter. For example, if "," is a delimiter and the option **Treat multiple delimiters as one** is selected, EViews will interpret ",," as a single delimiter. If the option is turned off, EViews will view this string as two delimiters surrounding a missing value.

Rectangular File Layout Options

To treat the ASCII file as a rectangular file, select the **File laid out as rectangle** option in the upper right-hand portion of the dialog. If the file is rectangular, EViews reads the file as a set of *lines*, with each new line denoting a new observation or a new series, depending on whether you are reading by column or by row. If you turn off the rectangular option, EViews treats the whole file as one long string separated by delimiters and carriage returns.

Knowing that a file is rectangular simplifies ASCII reading since EViews knows how many values to expect on a given line. For files that are not rectangular, you will need to be precise about the number of series or observations that are in your file. For example, suppose that you have a non-rectangular file that is ordered in columns and you tell EViews that there are four series in the file. EViews will ignore new lines and will read a new observation after reading every four values.

If the file is rectangular, you can tell EViews to skip columns and/or rows.

For example, if you have a rectangular file and you type 3 in the **Rows to skip** field, EViews will skip the first three rows of the data file. Note that you can only skip the *first* few rows or columns; you cannot skip rows or columns in the middle of the file.

Series Headers

This option tells EViews how many "cells" to offset as series name headers before reading the data in the file. The way that cell offsets are counted differs depending on whether the file is in rectangular form or not.

For files in rectangular form, the offsets are given by rows (for data in columns) or by columns (for data in rows). For example, suppose your data file looks as follows:

```
LABEL   P85   P75   RMT85   CS82   SS82   S82   ME84   REV84   REG   CL

1   33   27   288   13   24   49   2135   2836   1   1
2   19   15   139   14   12   41   957   2035   1   1
3   26   20   196   12   14   41   1530   6030   1   1
```

There is a one line (row) gap between the series name line and the data for the first observation. In this case, you should set the series header offset as 2, one for the series name line and one for the gap. If there were no gap, then the correct offset would instead be 1.

For files not in rectangular form, the offsets are given by the number of cells separated by the delimiters. For example, suppose you have a data file that looks as follows:

```
FTP     UEMP      MAN      LIC       GR     CLEAR      WM     NMAN      GOV      HE
  WE      HOM      ACC      ASR
260.35  11.0     455.5   178.15  215.98   93.4     558724. 538.1    133.9    2.98
117.18  8.60     39.17   306.18
269.80  7.0      480.2   156.41  180.48   88.5     538584. 547.6    137.6    3.09
134.02  8.90     40.27   315.16
```

The data are ordered in columns, but each observation is recorded in two lines, the first line for the first 10 series and the second line for the remaining 4 series.

It is instructive to examine what happens if you incorrectly read this file as a rectangular file with 14 series and a header offset of 2. EViews will look for the series names in the first line, will skip the second line, and will begin reading data starting with the third line, treating each line as one observation. The first 10 series names will be read correctly, but since EViews will be unable to find the remaining four names on the first line, the remaining series will be named SER01–SER04. The data will also be read incorrectly. For example, the first four observations for the series GR will be 215.9800, NA, 180.4800, and NA, since EViews treats each line as a new observation.

To read this data file properly, you should turn off the rectangle file option and set the header offset to 1. Then EViews will read, from left to right, the first 14 values that are separated by a delimiter or carriage return and take them as series names. This corresponds to the header offset of 1, where EViews looks to the number of series (in the upper left edit box) to determine how many cells to read per header offset. The next 14 observations are the first observations of the 14 series, and so on.

Miscellaneous Options

- **Quote with single ‘ not “**. The default behavior in EViews is to treat anything inside a pair of matching double quotes as one string, unless it is a number. This option treats anything inside a pair of matching single quotes as one string, instead of the double quotes. Since EViews does not support strings, the occurrence of a pair of matching double quotes will be treated as missing, unless the text inside the pair of double quotes may be interpreted as a number.

- **Drop strings—don’t make NA**. Any input into a numeric series that is not a number or delimiter will, by default, be treated as a missing observation. For example, “10b” and “90:4” will both be treated as missing values (unless Alphabetic characters or “:” are treated as delimiters). The **Drop strings** option will skip these strings instead of treating them as NAs.

  If you choose this option, the series names, which are strings, will also be skipped so that your series will be named using the EViews default names: “SER01”,

"SER02", and so on. If you wish to name your series, you should list the series names in the dialog.

Note that strings that are specified as missing observations in the **Text for NA** edit box will not be skipped and will be properly indicated as missing.

- **Numbers in ( ) are negative**. By default, EViews treats parentheses as strings. However, if you choose this option, numbers in parentheses will be treated as negative numbers and will be read accordingly.

- **Allow commas in numbers**. By default, commas are treated as strings unless you specify them as a delimiter. For example, "1,000" will be read as either NA (unless you choose the drop string option, in which case it will be skipped) or as two observations, 1 and 0 (if the comma is a delimiter). However, if you choose to **Allow commas in numbers**, "1,000" will be read as the number 1000.

- **Currency**. This option allows you to specify a symbol for currency. For example, the default behavior treats "$10"' as a string (which will either be NA or skipped) unless you specify "$" as a delimiter. If you enter "$" in the **Currency** option field, then "$10" will be read as the number 10.

  The currency symbol can appear either at the beginning or end of a number but not in the middle. If you type more than one symbol in the field, each symbol will be treated as a currency code. Note that currency symbols are case sensitive. For example, if the Japanese yen is denoted by the "Y" prefix, you should enter "Y", not "y".

- **Text for NA**. This option allows you to specify a code for missing observations. The default is NA. You can use this option to read data files that use special values to indicate missing values, *e.g.*, ".", or "-99".

  You can specify only one code for missing observations. The entire **Text for NA** string will be treated as the missing value code.

Examples

In these examples, we demonstrate the ASCII import options using example data files downloaded from the Internet. The first example file looks as follows:

```
 X  Y  Z  A   B    AA   BB
 5  3  10  -0.6008  NBA  10  8
10  3   7  -0.4837  NFL  7   2
 2  2   5   1.2467  NFL  0   0
12  0   3   0.1705  NBA  5   1
```

This is a cross-section data set, with seven series ordered in columns, each separated by a single space. Note that the B series takes string values, which will be replaced by NAs. If we type 7 series in the number of series field and use the default setting, EViews will correctly read the data.

By default, EViews checks the **Treat multiple delimiters as one** option even though the series are delimited by a single space. If you do not check this option, the last series BB will not be read. EViews will create a series named "SER01" and all data will be incorrectly imported. This strange behavior is caused by an extra space in the very first column of the data file, before the 1st and 3rd observations of the X series. EViews treats the very first space as a delimiter and looks for the first series data before the first extra space, which is missing. Therefore the first series is named SER01 with data NA, 10, NA, 12 and all other series are incorrectly imported.

To handle this case, EViews automatically ignores the delimiter before the first column data if you choose both the **Treat multiple delimiters as one** and the **File laid out as rectangle** options.

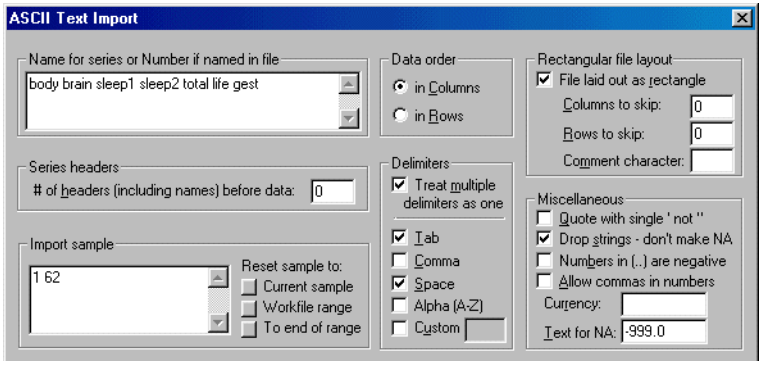
The top of the second example file looks like:

```
African elephant         6654.000 5712.000  -999.0  -999.0     3.3     38.6   645.0
African giant pouched rat   1.000    6.600     6.3     2.0     8.3      4.5    42.0
Arctic Fox                  3.385   44.500  -999.0  -999.0    12.5     14.0    60.0
Arctic ground squirrel       .920    5.700  -999.0  -999.0    16.5   -999.0    25.0
Asian elephant           2547.000 4603.000     2.1     1.8     3.9     69.0   624.0
Baboon                     10.550  179.500     9.1      .7     9.8     27.0   180.0
```

This is a cross-section data set, ordered in columns, with missing values coded as "-999.0". There are eight series, each separated by spaces. The first series is the ID name in strings.

If we use the EViews defaults, there will be problems reading this file. The spaces in the ID description will generate spurious NA values in each row, breaking the rectangular format of the file. For example, the first name will generate two NAs, since "African" is treated as one string, and "elephant" as another string.

You will need to use the **Drop strings** option to skip all of the strings in your data so that you don't generate NAs. Fill out the ASCII dialog as follows:



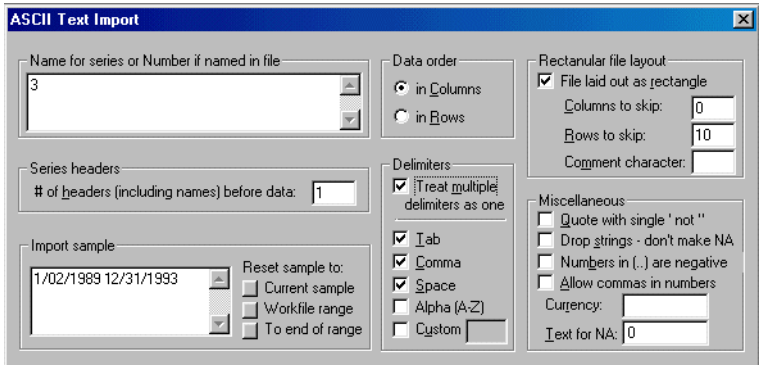
Note the following:

- Since we skip the first string series, we list only the remaining seven series names.
- There are no header lines in the file, so we set the offset to 0.
- If you are not sure whether the delimiter is a space or tab, mark both options. You should treat multiple delimiters as one.
- **Text for NA** should be entered exactly as it appears in the file. For this example, you should enter "`-999.0`", not "`-999`".

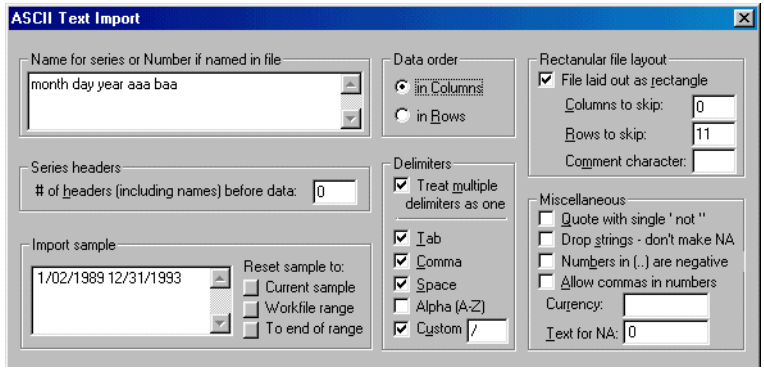
The third example is a daily data file that looks as follows:

```
"Daily Corporate Bond Yields"
"Jan. 1, 1989 to Dec. 31, 1993"

"Source:  Federal Reserve Statistical Release H.15"
"Board of Governors of the Federal Reserve System"

"DATE    =  MM/DD/YY FORMAT"
"AA      =  MOODYS SEASONED AA"
"BAA     =  MOODYS SEASONED BAA"

"DATE",       "AA",      "BAA"
01/02/89        0          0
01/03/89       9.88      10.71
01/04/89       9.86      10.71
01/05/89       9.89      10.75
```

This file has 10 lines of data description, line 11 is the series name header, and the data begin in line 12. The data are ordered in columns in rectangular form with missing values coded as a "0". To read these data, you can instruct EViews to skip the first 10 rows of the rectangular file, and read three series with the names in the file, and NAs coded as "0".



The only problem with this method is that the DATE series will be filled with NAs since EViews treats the entry as a string (because of the "/" in the date entry). You can avoid this problem by identifying the slash as a delimiter using the **Custom** edit box. The first column will now be read as three distinct series since the two slashes are treated as delimiters. Therefore, we modify the option settings as follows:



Note the changes to the dialog entries:

- We now list five series names. We cannot use the file header since the line only contains three names.
- We skip 11 rows with no header offset since we want to skip the name header line.
- We specify the slash "/" as an additional delimiter in the **Custom** option field.

The month, day, and year will be read as separate series and can be used as a quick check of whether the data have been read correctly.

Chapter 6. Working with Data

In the following discussion, we describe EViews' powerful language for using numeric expressions and generating and manipulating the data in series and groups. We first describe the fundamental rules for working with mathematical expressions in EViews, and then describe how to use these expressions in working with series and group data.

More advanced tools for working with numeric data, and objects for working with different kinds of data are described in Chapter 7, "Working with Data (Advanced)".

Numeric Expressions

One of the most powerful features of EViews is the ability to use and to process mathematical expressions. EViews contains an extensive library of built-in operators and functions that allow you to perform complicated mathematical operations on your data with just a few keystrokes. In addition to supporting standard mathematical and statistical operations, EViews provides a number of specialized functions for automatically handling the leads, lags and differences that are commonly found in time series data.

An EViews expression is a combination of numbers, series names, functions, and mathematical and relational operators. In practical terms, you will use expressions to describe all mathematical operations involving EViews objects.

As in other programs, you can use these expressions to calculate a new series from existing series, to describe a sample of observations, or to describe an equation for estimation or forecasting. However, EViews goes far beyond this simple use of expressions by allowing you to use expressions virtually anywhere you would use a series. We will have more on this important feature shortly, but first, we describe the basics of using expressions.

Operators

EViews expressions may include operators for the usual arithmetic operations. The operators for addition (+), subtraction (-), multiplication (*), division (/) and raising to a power (^) are used in standard fashion so that:

```
5 + 6 * 7.0 / 3
7 + 3e-2 / 10.2345 + 6 * 10^2 + 3e3
3^2 - 9
```

are all valid expressions. Notice that explicit numerical values may be written in integer, decimal, or scientific notation.

In the examples above, the first expression takes 5 and adds to it the product of 6 and 7.0 divided by 3 (5 + 14 = 19); the last expression takes 3 raised to the power 2 and subtracts 9 (9 – 9 = 0). These expressions use the order of evaluation outlined below.

The "-" and "+" operators are also used as the unary minus (negation) and unary plus operators. It follows that:

```
2-2
-2+2
2++++++++++++-2
2---2
```

all yield a value of 0.

EViews follows the usual order in evaluating expressions from left to right, with operator precedence order as follows (from highest precedence to lowest):

- unary minus (-), unary plus (+)
- exponentiation (^)
- multiplication (*), division (/)
- addition (+), subtraction (-)
- comparison (<, >, <=, >=, =)
- and, or

The last two sets of operators are used in logical expressions.

To enforce a particular order of evaluation, you can use parentheses. As in standard mathematical analysis, terms which are enclosed in parentheses are treated as a subexpression and evaluated first, from the innermost to the outermost set of parentheses. We strongly recommend the use of parentheses when there is any possibility of ambiguity in your expression.

To take some simple examples,

- -1^2, evaluates to (–1)^2 = 1 since the unary minus is evaluated prior to the power operator.
- -1 + -2 * 3 + 4, evaluates to –1 + –6 + 4 = –3. The unary minus is evaluated first, followed by the multiplication, and finally the addition.
- (-1 + -2) * (3 + 4), evaluates to –3 * 7 = –21. The unary minuses are evaluated first, followed by the two additions, and then the multiplication.
- 3*((2+3)*(7+4) + 3), evaluates to 3 * (5*11 + 3) = 3 * 58 = 174.

A full listing of operators is presented in Appendix D, "Operator and Function Reference", on page 573 of the *Command and Programming Reference*.

Series Expressions

Much of the power of EViews comes from the fact that expressions involving series operate on every observation, or element, of the series in the current sample. For example, the series expression:

```
2*y + 3
```

tells EViews to multiply every sample value of Y by 2 and then to add 3. We can also perform operations that work with multiple series. For example:

```
x/y + z
```

indicates that we wish to take every observation for X and divide it by the corresponding observation on Y, and add the corresponding observation for Z.

Series Functions

EViews contains an extensive library of built-in functions that operate on all of the elements of a series in the current sample. Some of the functions are "element functions" which return a value for each element of the series, while others are "summary functions" which return scalars, vectors or matrices, which may then be used in constructing new series or working in the matrix language (see Chapter 3, "Matrix Language", on page 23 of the *Command and Programming Reference* for a discussion of scalar, vector and matrix operations).

Most function names in EViews are preceded by the `@`-sign. For example, `@mean` returns the average value of a series taken over the current sample, and `@abs` takes the absolute value of each observation in the current sample.

All element functions return NAs when any input value is missing or invalid, or if the result is undefined. Functions which return summary information generally exclude observations for which data in the current sample are missing. For example, the `@mean` function will compute the mean for those observations in the sample that are non-missing.

There is an extensive set of functions that you may use with series:

- A list of mathematical functions is presented in Appendix D, "Operator and Function Reference", on page 573 of the *Command and Programming Reference*.
- Workfile functions that provide information about observations identifiers or allow you to construct time trends are described in Appendix E, "Workfile Functions", on page 589 of the *Command and Programming Reference*.

- Functions for working with strings and dates are documented in "String Function Summary" on page 129 of the *Command and Programming Reference* and "Date Function Summary" on page 152 of the *Command and Programming Reference*.

The remainder of this chapter will provide additional examples of expressions involving functions.

Series Elements

At times, you may wish to access a particular observation for a series. EViews provides you with a special function, `@elem`, which allows you to use a specific value of a series.

`@elem` takes two arguments: the first argument is the name of the series, and the second is the date or observation identifier.

For example, suppose that you want to use the 1980Q3 value of the quarterly series Y, or observation 323 of the undated series X. Then the functions:

```
@elem(y, 1980Q3)
@elem(x, 323)
```

will return the values of the respective series in the respective periods.

Numeric Relational Operators

Relational comparisons may be used as part of a mathematical operation, as part of a sample statement, or as part of an if-condition in programs.

A numeric relational comparison is an expression which contains the " = " (equal), " > = " (greater than or equal), " < = " (less than or equal), " < > " (not equal), " > " (greater than), or " < " (less than) comparison operators. These expressions generally evaluate to TRUE or FALSE, returning a 1 or a 0, depending on the result of the comparison.

Comparisons involving strings are discussed in "String Relational Operators" beginning on page 121 of the *Command and Programming Reference*.

Note that EViews also allows relational comparisons to take the value "missing" or NA, but for the moment, we will gloss over this point until our discussion of missing values (see "Missing Values" on page 134).

We have already seen examples of expressions using relational operators in our discussion of samples and sample objects. For example, we saw the sample condition:

```
incm > 5000
```

which allowed us to select observations meeting the specified condition. This is an example of a relational expression—it is TRUE for each observation on INCM that exceeds 5000; otherwise, it is FALSE.

As described above in the discussion of samples, you may use the "and" and "or" conjunction operators to build more complicated expressions involving relational comparisons:

```
(incm>5000 and educ>=13) or (incm>10000)
```

It is worth emphasizing the fact that EViews uses the number 1 to represent TRUE and 0 to represent FALSE. This internal representation means that you can create complicated expressions involving logical subexpressions. For example, you can use relational operators to recode your data:

```
0*(inc<100) + (inc>=100 and inc<200) + 2*(inc>=200)
```

which yields 0 if INC < 100, 1 if INC is greater than or equal to 100 and less than 200, and 2 for INC greater than or equal to 200.

The equality comparison operator "=" requires a bit more discussion, since the equal sign is used both in assigning values and in comparing values. We consider this issue in greater depth when we discuss creating and modifying series (see "Series" on page 137). For now, note that if used in an expression:

```
incm = 2000
```

evaluates to TRUE if INCOME is exactly 2000, and FALSE, otherwise.

Leads, Lags, and Differences

It is easy to work with lags or leads of your series. Simply use the series name, followed by the lag or lead enclosed in parentheses. Lags are specified as negative numbers and leads as positive numbers so that,

```
income(-4)
```

is the fourth lag of the income series, while:

```
sales(2)
```

is the second lead of sales.

While EViews expects lead and lag arguments to be integers, there is nothing to stop you from putting non-integer values in the parentheses. EViews will automatically convert the number to an integer; you should be warned, however, that the conversion behavior is not guaranteed to be systematic. If you must use non-integer values, you are strongly encouraged to use the `@round`, `@floor`, or `@ceil` functions to control the lag or lead behavior.

In many places in EViews, you can specify a range of lead or lag terms. For example, when estimating equations, you can include expressions of the form:

```
income(-1 to -4)
```

to represent all of the INCOME lags from 1 to 4. Similarly, the expressions:

```
sales sales(-1) sales(-2) sales(-3) sales(-4)
sales(0 to -4)
sales(to -4)
```

are equivalent methods of specifying the level of SALES and all lags from 1 to 4.

EViews also has several built-in functions for working with difference data in either levels or in logs. The "D" and "DLOG" functions will automatically evaluate the differences for you. For example, instead of taking differences explicitly,

```
income - income(-1)
log(income) - log(income(-1))
```

you may use the equivalent expressions,

```
d(income)
dlog(income)
```

You can take higher order differences by specifying the difference order. For example, the expressions:

```
d(income,4)
dlog(income,4)
```

represent the fourth-order differences of INCOME and log(INCOME).

If you wish to take seasonal differences, you should specify both the ordinary, and a seasonal difference term:

```
d(income,1,4)
dlog(income,1,4)
```

These commands produce first order differences with a seasonal difference at lag 4. If you want only the seasonal difference, specify the ordinary difference term to be 0:

```
d(income,0,4)
dlog(income,0,4)
```

Mathematical details are provided in Appendix D, "Operator and Function Reference", on page 573 of the *Command and Programming Reference*.

Missing Values

Occasionally, you will encounter data that are not available for some periods or observations, or you may attempt to perform mathematical operations where the results are unde-

fined (*e.g.*, division by zero, log of a negative number). EViews uses the code NA (not available) to represent these missing values.

For the most part, you need not worry about NAs. EViews will generate NAs for you when appropriate, and will automatically exclude observations with NAs from statistical calculations. For example, if you are estimating an equation, EViews will use the set of observations in the sample that have no missing values for the dependent and all of the independent variables.

There are, however, a few cases where you will need to work with NAs, so you should be aware of some of the underlying issues in the handling of NAs.

First, when you perform operations using multiple series, there may be alternative approaches for handling NAs. EViews will usually provide you with the option of *casewise exclusion* (common sample) or *listwise exclusion* (individual sample). With casewise exclusion, only those observations for which *all* of the series have non-missing data are used. This rule is always used, for example, in equation estimation. For listwise exclusion, EViews will use the maximum number of observations possible for each series, excluding observations separately for each series in the list of series. For example, when computing descriptive statistics for a group of series, you have the option to use a different sample for each series.

If you must work directly with NAs, just keep in mind that EViews NAs observe all of the rules of IEEE NaNs. This means that performing mathematical operations on NAs will generate missing values. Thus, each of the following expressions will generate missing values:

```
@log(-abs(x))
1/(x-x)
(-abs(x))^(1/3)
3*x + NA
exp(x*NA)
```

For the most part, comparisons involving NA values propagate NA values. For example, the commands:

```
series y = 3
series x = NA
series equal = (y = x)
series greater = (y > x)
```

will create series EQUAL and GREATER that contain NA values, since the comparison between observations in a series involving an NA yields an NA.

Note that this behavior differs from EViews 4.1 and earlier in which NAs were treated as ordinary values for purposes of equality (“ = ”) and inequality (“ < > ”) testing. In these

versions of EViews, the comparison operators " = " and " < > " always returned a 0 or a 1. The change in behavior was deemed necessary to support the use of string missing values. In all versions of EViews, comparisons involving ordering (" > ", " < ", " < = ", " > = ") propagate NAs.

It is still possible to perform comparisons using the previous methods. One approach is to use the special functions @EQNA and @NEQNA for performing equality and strict inequality comparisons without propagating NAs. For example, you may use the commands:

```
series equal1 = @eqna(x, y)
series nequal = @neqna(x, y)
```

so that NAs in either X or Y are treated as ordinary values for purposes of comparison. Using these two functions, EQUAL1 will be filled with the value 0, and NEQUAL will be filled with the value 1. Note that the @EQNA and @NEWNA functions do not compare their arguments to NA, but rather facilitate the comparison of values so that the results are guaranteed to be 0 or 1. See also "Version 4 Compatibility Mode" on page 97 of the *Command and Programming Reference* for settings that enable the previous behavior for element comparisons in programs.

To test whether individual observations in a series are NAs, you may use the @ISNA function. For example,

```
series isnaval = @isna(x)
```

will fill the series ISNAVAL with the value 1, since each observation in X is an NA.

There is one special case where direct comparison involving NAs does not propagate NAs. If you test equality or strict inequality against the literal NA value:

```
series equal2 = (x = NA)
series nequal2 = (y <> NA)
```

EViews will perform a special test against the NA value without propagating NA values. Note that these commands are equivalent to the comparisons involving the special functions:

```
series equal3 = @eqna(x, NA)
series nequal3 = @neqna(y, NA)
```

If used in a mathematical operation, a relational expression resulting in an NA is treated as an ordinary missing value. For example, for observations where the series X contains NAs, the mathematical expression

```
5*(x>3)
```

will yield NAs. However, if the relational expression is used as part of a sample or IF-statement, NA values are treated as FALSE.

```
smpl 1 1000 if x>y
smpl 1 1000 if x>y and not @isna(x) and not @isna(y)
```

are equivalent since the condition `x>3` implicitly tests for NA values. One consequence of this behavior is that:

```
smpl 1 1000 if x<NA
```

will result in a sample with no observations since less-than tests involving NAs yield NAs.

Very early versions of EViews followed the IEEE rules for missing data with one important exception. In EViews 2 and earlier, multiplying any number by zero (including NAs) yielded a zero. In subsequent versions, the value NA times zero equals NA. Thus, an earlier recommended method of recoding (replacing) NA values in the series X no longer worked so that the command for replacing NA values with the values in Y:

```
x = (x<>na)*x + (x=na)*y
```

works in EViews 2, but does not work subsequent versions. The `@nan` function has been provided for this purpose.

```
x = @nan(x,y)
```

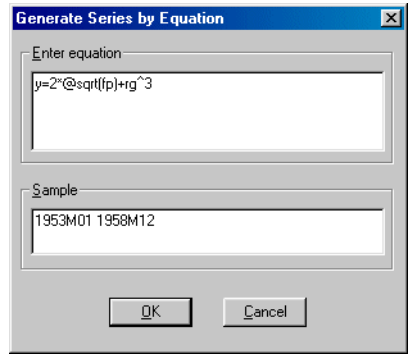
recodes NA values of X to take the values in the series Y. See "Basic Mathematical Functions" on page 575 of the *Command and Programming Reference*.

Series

One of the primary uses of expressions is to generate new series from existing data or to modify the values in an existing series. Used in combination with samples, expressions allow you to perform sophisticated transformations of your data, saving the results in new or existing series objects.

The current discussion focuses on the basic numeric series object. Users who wish to work with alphanumeric or advanced series features should see Chapter 7, "Working with Data (Advanced)", on page 149 and Chapter 8, "Series Links", on page 177.

To create or modify a series, select **Quick/Generate Series...** or click on the **Genr** button on the workfile toolbar. EViews opens a window prompting you for additional information.



You should enter the assignment statement in the upper edit box, and the relevant sample period in the lower edit box.

The assignment statement is actually an implicit loop over observations. Beginning with the first observation in the sample, EViews will evaluate the assignment statement for each included observation.

Basic Assignment

You can type the series name, followed by an equal sign and then an expression. For every element of the sample, EViews will evaluate the expression on the right-hand side of the equality, and assign the value to the *destination series* on the left-hand side, creating the series if necessary.

For example, if there is no series named Y,

```
y = 2*x + 37*z
```

will first create the Y series and fill it with NAs. Then, for every observation in the current sample, EViews will fill each element of the Y series with the value of the expression. If Y does exist, EViews will only replace Y values in the current sample with the value of the expression. All observations not in the sample will be unchanged.

One special form of assignment occurs when the right-hand side of the assignment statement is a constant expression:

```
y = 3
y = 37 * 2 + 3
```

EViews will simply assign the value of the constant to all of the observations in the sample.

Using Samples

By modifying the sample of observations used in assignment, you can splice together series using multiple **Genr** commands. For example, if we enter three **Genr** commands with different samples: first

Upper window: `y = z`

Lower window: `@all if z<=1 and z>-1`

followed by a **Genr** with,

Upper window: `y = -2 + 3*z`

Lower window: `if z>1`

and finally,

Upper window: `y = -.9 + .1*z`

Lower window: `if z<=-1`

we can generate Y as a piecewise linear function of the series Z. Note that the "@ALL" is implicit in the latter two assignments.

While it is possible to perform these types of operations using loops and IF-statements (see the *Command and Programming Reference*), we strongly urge you to use **Genr** and sample statements where possible, since the latter approach is much more efficient.

Dynamic Assignment

Since EViews evaluates the assignment expression for each observation in the sample, you can perform dynamic assignment by using lagged values of the destination series on the right side of the equality. For example, suppose we have an annual workfile that ranges from 1945 to 1997. Then if we enter:

Upper window: `y = y + y(-1)`

Lower window: `1946 1997`

EViews will replace the Y series with the cumulative sum of Y. We begin with 1946, since we do not want to transform the first value in the workfile. Then for each period, EViews will take the current value of Y and add it to the lagged value of Y. The assignment is dynamic because as we successively move on to the next period, the lagged value of Y contains the cumulative sum.

Note that this procedure destroys the original data. To create a new series with the cumulative sums, you will have to perform the assignment in two steps, first making a copy of the original series, and then performing the dynamic assignment.

Implicit Assignment

You can make an implicit assignment by putting a simple formula on the left-hand side of the equal sign. EViews will examine your expression and select, as the destination series, the first valid series name on the left-hand side of the equality. Then for every observation in the sample, EViews will assign values using the implicit relationship. For example, if you enter:

```
log(y) = x
```

EViews will treat Y as the destination series, and evaluate `y=exp(x)` for every observation in the sample.

The following are examples of valid assignment statements where Y is the destination series:

```
1/y = z
log(y/x)/14.14 = z
log(@inv(y)*x) = z
2+y+3*z = 4*w
d(y) = nrnd
```

In general, EViews can solve for, or *normalize,* equations that use the following on the left-hand side of the equality: `+`, `–`, `*`, `/`, `^`, `log()`, `exp()`, `sqr()`, `d()`, `dlog()`, `@inv()`.

Since **Genr** is not a general equation solver, there will be situations in which EViews cannot normalize your equation. You cannot, for example, use the assignment statement:

```
@tdist(y, 3) = x
```

since `@tdist` is not one of the functions that EViews knows how to invert. Similarly, EViews cannot solve for equations where the destination series appears more than once on the left side of the equality. For example, EViews cannot solve the equation:

```
x + 1/x = 5
```

In both cases, EViews will display the error message "Unable to normalize equation".

Note that the destination series can appear on both sides of the equality. For example:

```
log(x) = x
```

is a legal assignment statement. EViews will normalize the expression and perform the assignment

```
x = exp(x)
```

so that X will be assigned the exponential of the original value of X. EViews will *not* solve for the values of X satisfying the equality "LOG(X) = X".

Using the Command Window

You can create series and assign values from the command window. First, set the workfile sample using the `smpl` statement, then enter the assignment statement.

There are alternative forms for the assignment statement. First, if the series does not exist, you must use either the `series` or the `genr` keyword, followed by the assignment expression. The two statements:

```
series y = exp(x)
genr y = exp(x)
```

are equivalent methods of generating the series Y. Once the series has been created, subsequent assignment statements do not require the `series` or the `genr` keyword:

```
smpl @all
series y = exp(x)
smpl 1950 1990 if y>300
y = y/2
```

This set of commands first sets the series to equal EXP(X) for all observations, then assigns the values Y/2 for the subset of observations from 1950 to 1990 if Y > 300.

Auto-series

Another important method of working with expressions is to use an expression *in place of* a series. EViews' powerful tools for expression handling allow you to substitute expressions virtually any place you would use a series—as a series object, as a group element, in equation specifications and estimation, and in models.

We term expressions that are used in place of series as *auto-series*, since the transformations in the expressions are *automatically* calculated without an explicit assignment statement.

Auto-series are most useful when you wish to see the behavior of an expression involving one ore more series, but do not want to keep the transformed series, or in cases where the underlying series data change frequently. Since the auto-series expressions are automatically recalculated whenever the underlying data change, they are never out-of-date.

See "Auto-Updating Series" on page 149 for a more advanced method of handling series and expressions.

Creating Auto-series

It is easy to create and use an auto-series—anywhere you might use a series name, simply enter an EViews expression. For example, suppose that you wish to plot the log of CP against time for the period 1953M01 to 1958M12. There are two ways in which you might plot these values.

One way to plot these values is to generate an ordinary series, as described earlier in "Basic Assignment" on page 138, and then to plot its values. To generate an ordinary series

containing the log of CP, say with the name LOGCP, select **Quick/Generate series...** from the main menu, and enter,

```
logcp = log(cp)
```

or type the command,

```
series logcp = log(cp)
```

in the command window. EViews will evaluate the expression LOG(CP) for the current values of CP, and will place these values into the series LOGCP. To view a line graph view of the series, open the series LOGCP and select **View/Graph/Line**.

Note that the values of the ordinary series LOGCP will not change when CP is altered. If you wish to update the values in LOGCP to reflect subsequent changes in CP, you will need to issue another `series` or `genr` assignment statement.

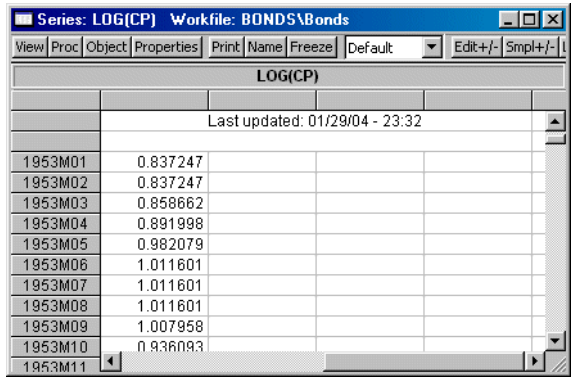
Alternatively, you may create and use an auto-series by clicking on the **Show** button on the toolbar, or selecting **Quick/Show...** and entering the command,

```
log(cp)
```

or by typing

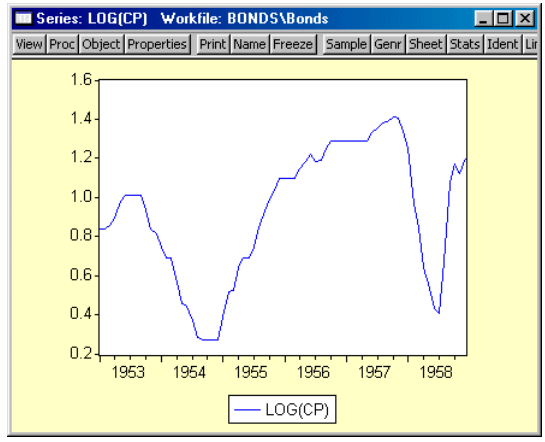
```
show log(cp)
```

in the command window. EViews will open a series window in spreadsheet view:



Note that in place of an actual series name, EViews substitutes the expression used to create the auto-series.

An auto-series may be treated as a standard series window so all of the series views and procedures are immediately available. To display a time series graph of the LOG(CP) auto-series, simply select **View/Graph/Line** from the series window toolbar:

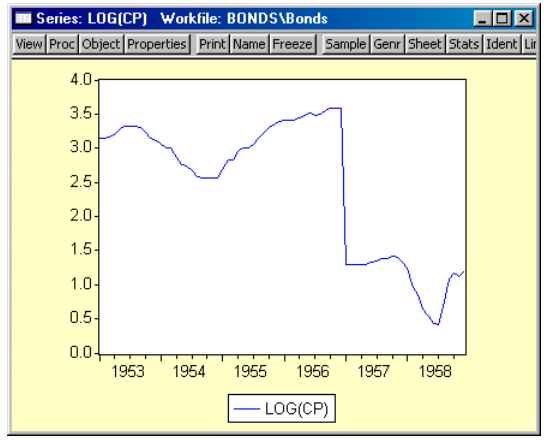


All of the standard series views and procedures are also accessible from the menus.

Note that if the data in the CP series are altered, the auto-series will reflect these changes. Suppose, for example, that we take the first four years of the CP series, and multiply theme by a factor of 10:

```
smpl 1953m01 1956m12
cp = cp*10
smpl 1953m01 1958m12
```

The auto-series graph will automatically change to reflect the new data:

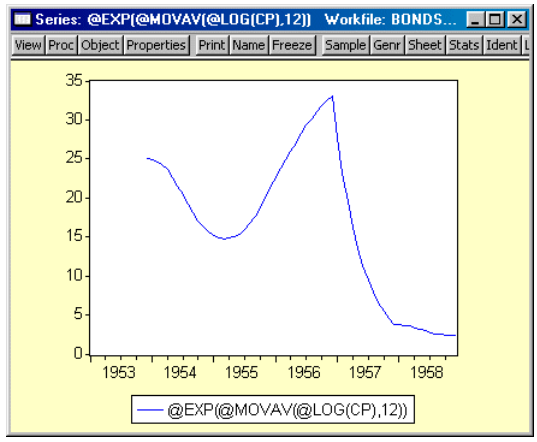


In contrast, the values of the ordinary series LOGCP are not affected by the changes in the CP data.

Similarly, you may use an auto-series to compute a 12-period, backward-looking, geometric moving average of the *updated* CP data. The command:

```
show @exp(@movav(@log(cp),12))
```

will display the auto-series containing the geometric moving average:



Naming an Auto-series

The auto-series is deleted from your computer memory when you close the series window containing the auto-series. For more permanent expression handling, you may convert the auto-series into an auto-updating series that will be kept in the workfile, by assigning a name to the auto-series.

Simply click on the **Name** button on the series toolbar, or select **Object/Name...** from the main menu, and provide a name. EViews will create an auto-updating series with that name in the workfile, and will assign the auto-series expression as the formula used in updating the series. For additional details, see "Auto-Updating Series" on page 149.

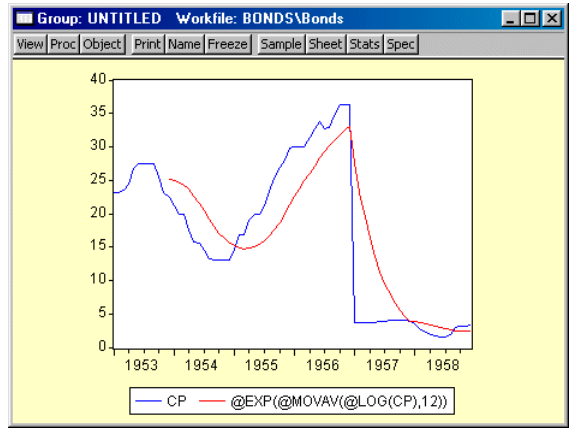
Using Auto-series in Groups

One of the more useful ways of working with auto-series is to include them in a group. Simply create the group as usual, using an expression in place of a series name, as appropriate. For example, if you select **Object/New Object.../Group**, and enter:

```
cp @exp(@movav(@log(cp),12))
```

you will create a group containing two series: the ordinary series CP, and the auto-series representing the geometric moving average. We may then use the group object graphing routines to compare the original series with the smoothed series:

"Groups" on page 145 below describes other useful techniques for working with auto-series.



Using Auto-Series in Estimation

One method of using auto-series in estimation is to allow expressions as right-hand side variables. Thus, you could estimate an equation with `log(x)` or `exp(x+z)` as an explanatory variable.

EViews goes a step beyond this use of auto-series, by allowing you to use auto-series as the dependent variable in estimation. Thus, if you want to regress the log of Y on explanatory variables, you don't have to create a new variable LOGY. Instead, you can use the expression `log(y)` as your dependent variable.

When you forecast using an equation with an auto-series dependent variable, EViews will, if possible, forecast the untransformed dependent variable and adjust the estimated confidence interval accordingly. For example, if the dependent variable is specified as `log(y)`, EViews will allow you to forecast the level of Y, and will compute the asymmetric confidence interval. See Chapter 18, "Forecasting from an Equation", on page 543 for additional details.

Groups

EViews provides specialized tools for working with groups of series that are held in the form of a group object. In "Importing Data" on page 105 we used groups to import data from spreadsheets into existing workfiles. Briefly, a group is a collection of one or more series identifiers or expressions. Note that a group does not contain the data in the individual series, only references to the data in the series.

To create a group, select **Object/New Object.../Group** and fill in the dialog with names of series and auto-series. Or you may select **Show** from the workfile toolbar and fill out the dialog. Alternatively, type the command `group` in the command window, followed by a name to be given to the group and then the series and auto-series names:

```
group macrolist gdp invest cons
```

creates the group MACROLIST containing the series GDP, INVEST and CONS. Similarly,

```
group altlist log(gdp) d(invest) cons/price
```

creates the group ALTLIST containing the log of the series GDP, the first difference of the series INVEST, and the CONS series divided by the PRICE series.

There are a few features of groups that are worth keeping in mind:

- A group is simply a list of series identifiers. It is not a copy of the data in the series. Thus, if you change the data for one of the series in the group, you will see the changes reflected in the group.
- If you delete a series from the workfile, the series identifier will be maintained in all groups. If you view the group spreadsheet, you will see a phantom series containing NA values. If you subsequently create or import the series, the series values will be restored in all groups.
- Renaming a series changes the reference in every group containing the series, so that the newly named series will still be a member of each group.
- There are many routines in EViews where you can use a group name in place of a list of series. If you wish, for example, to use X1, X2 and X3 as right-hand side variables in a regression, you can instead create a group containing the series, and use the group in the regression.

We describe groups in greater detail in Chapter 12, “Groups”, on page 363.

Accessing Individual Series in a Group

Groups, like other EViews objects, contain their own views and procedures. For now, note that you can access the individual elements of a named group as individual series.

To refer the n-th series in the group, simply append “(n)” to the group name. For example, consider the MACROLIST group, defined above. The expression MACROLIST(1) may be used to refer to GDP and MACROLIST(2) to refer to INVEST.

You can work with MACROLIST(1) as though it were any other series in EViews. You can display the series by clicking on the **Show** button on the toolbar and entering MACROLIST(1). You can include GDP in another group directly or indirectly. A group which contains:

```
macrolist(1) macrolist(2)
```

will be identical to a group containing

```
gdp invest
```

We can also use the individual group members as part of expressions in generating new series:

```
series realgdp = macrolist(1)/price
series y = 2*log(macrolist(3))
```

or in modifying the original series:

```
series macrolist(2) = macrolist(2)/price
```

Note that in this latter example the `series` keyword is required, despite the fact that the INVEST series already exists. This is true whenever you access a series as a member of a group.

Other tools allow you to retrieve the number of series in a group using the "@COUNT" group data member:

```
scalar numgroup = macrolist.@count
```

To retrieve the names of each of the series, you may use the group data member "@SERIESNAME". These tools are described in greater detail in "Group Data Members" on page 164 of the *Command and Programming Reference*.

An Illustration

Auto-series and group processing provides you with a powerful set of tools for working with series data. As we saw above, auto-series provide you with dynamic updating of expressions. If we use the auto-series expression:

```
log(y)
```

the result will be automatically updated whenever the contents of the series Y changes.

A potential drawback of using auto-series is that expressions may be quite lengthy. For example, the two expressions:

```
log(gdp)/price + d(invest) * (cons + invest)
12345.6789 * 3.14159 / cons^2 + dlog(gdp)
```

are not suited to use as auto-series if they are to be used repeatedly in other expressions.

You can employ group access to make this style of working with data practical. First, create groups containing the expressions:

```
group g1 log(gdp)/price+d(invest)*(cons+invest)
group g2 12345.6789*3.14159/cons^2+dlog(gdp)
```

If there are spaces in the expression, the entire contents should be enclosed in parentheses.

You can now refer to the auto-series as G1(1) and G2(1). You can go even further by combining the two auto-series into a single group:

```
group myseries g1(1) g2(1)
```

and then referring to the series as MYSERIES(1) and MYSERIES(2). If you wish to skip the intermediate step of defining the subgroups G1 and G2, make certain that there are no spaces in the subexpression or that it is enclosed in parentheses. For example, the two expressions in the group ALTSERIES,

```
group altseries (log(gdp)/price) 3.141*cons/price
```

may be referred to as ALTSERIES(1) and ALTSERIES(2).

Scalars

Scalar objects are different from series and groups in that they hold a single number instead of data for each observation in the sample. In addition, scalar objects have no window views, and may only be used in calculations or displayed on the status line. Scalars are created by commands of the form:

```
scalar scalar_name = number
```

where you assign a number to the scalar name. The number may be an expression or special functions that return a scalar.

To examine the contents of a scalar, you may enter the command `show`, followed by the name of the scalar. EViews will display the value of the scalar in the status line at the bottom of the EViews window, in the left-hand corner of the status line. For example:

```
scalar logl1 = eq1.@logl
show logl1
```

stores the log likelihood value of the equation object named EQ1 in a scalar named LOGL1, and displays the value in the status line. Likewise, double clicking on the scalar name in the workfile window displays the value in the status line.

Chapter 7. Working with Data (Advanced)

In addition to the basic tools for working with numeric data outlined in Chapter 6, "Working with Data", EViews provides additional tools and objects for more advanced data handling, or for working with different kinds of data.

Auto-Updating Series

One of the most powerful features of EViews is the ability to use a series expression in place of an existing series. These expressions generate auto-series in which the expression is calculated when in use, and automatically recalculated whenever the underlying data change, so that the values are never out of date.

Auto-series are designed to be discarded after use. The resulting downside to auto-series is that they are quite transitory. You must, for example, enter the expression wherever it is used; for example, you must type "LOG(X)" every time you wish to use an auto-series for the logarithm of X. For a single use of a simple expression, this requirement may not be onerous, but for more complicated expressions used in multiple settings, repeatedly entering the expression quickly becomes tedious.

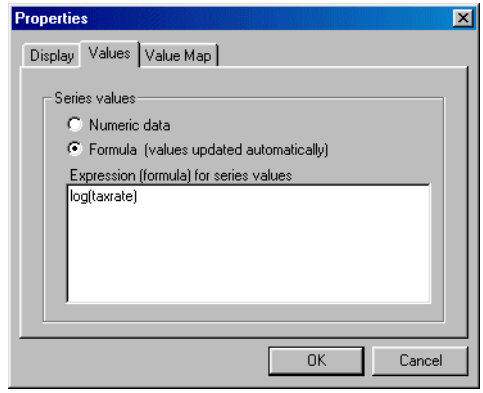
For more permanent series expression handling, EViews provides you with the ability to define a series or alpha object that uses a formula. The resulting auto-updating series is simply an EViews numeric series or alpha series that is defined, not by the values currently in the object, but rather by an expression that is used to compute the values. In most respects, an auto-updating series may simply be thought of as a named auto-series. Indeed, naming an auto-series is one way to create an auto-updating series.

The formula used to define an auto-series may contain any of the standard EViews series expressions, and may refer to series data in the current workfile page, or in EViews databases on disk. It is worth emphasizing that in contrast with link objects, which also provide dynamic updating capabilities, auto-updating series are designed to work with data in a single workfile page.

Auto-updating series appear in the workfile with a modified version of the series or alpha series icon, with the numeric series icon augmented by an " = " sign to show that it depends upon a formula.

Defining an Auto-Updating Series

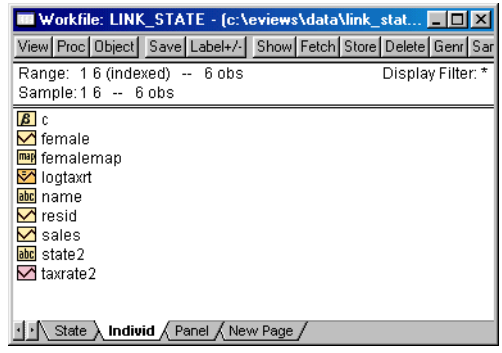
Using the Dialog



To turn a series into an auto-updating series, you will assign an expression to the series and tell EViews to use this expression to determine the series values. Simply click on the **Properties** button on the series or alpha series toolbar, or select **View/Properties...** from the main menu, then select the **Values** tab.

There are two radio buttons which control the values that will be placed in the numeric or alpha series ("Alpha Series" beginning on page 153). The default setting is either **Numeric data** or **Alphanumeric (text) data** (depending on the series type) in which the series is defined by the values currently in the series; this is the traditional way that one thinks of defining a numeric or alpha series.

If instead you select **Formula**, enter a valid series expression in the dialog box, and click on **OK**, EViews will treat the series as an auto-updating series and will evaluate the expression, putting the resulting values in the series. Auto-updating numeric series appear with a new icon in the workfile—a slightly modified version of the standard series icon, featuring the series line with an extra equal sign, all on an orange background.



In this example, we instruct EViews that the existing series LOGTAXRT should be an auto-updating series that contains the natural logarithm of the TAXRATE2 series. As with an auto-series expression, the values in LOGTAXRT will never be out of date since they will change to reflect changes in TAXRATE2. In contrast to an auto-series, however, LOGTAXRT is a permanent series in the workfile which may be used like any other series.

You may, at any time, change an auto-updating series into an standard numeric series by bringing up the **Values** page of the **Properties** dialog, and clicking on the **Numeric data** setting. EViews will define then define the series by its current values. In this way you may freeze the formula series values at their existing values, a procedure that is equivalent to performing a standard series assignment using the provided expression.

Note that once an expression is entered as a formula in a series, EViews will keep the definition even if you specify the series by value. Thus, you make take a series that has previously been frozen, and return it to auto-updating by selecting **Formula** definition.

Issuing a Command

To create an auto-updating series using commands, you should use the formula keyword, `frml`, followed by an assignment statement. The following example creates a series named LOW that uses a formula to compute values. The auto-updating series takes the value 1 if either INC is less than or equal to 5000 or EDU is less than 13, and takes the value 0 otherwise:

```
frml low = inc<=5000 or edu<13
```

LOW is now an auto-updating series that will be reevaluated whenever INC or EDU change.

You may also define auto-updating alpha series using the `frml` keyword. If FIRST_NAME and LAST_NAME are alpha series, then the declaration:

```
frml full_name = first_name + " " + last_name
```

creates an auto-updating alpha series, FULL_NAME.

The same syntax should be used when you wish to apply a formula to an existing series.

```
series z = rnd
frml z =(x+y)/2
```

makes Z an auto-updating series that contains the average of series X and Y. Note that the previous values of Z are replaced, and obviously lost. Similarly, we may first define an alpha series and then apply an updating formula:

```
alpha a = "initial value"
frml a = @upper(first_name)
```

You may not, however, apply an alpha series expression to a numeric series, or vice versa. Given the series Z and A defined above, the following two statements:

```
frml z = @upper(first_name)
frml a = (x+y)/2
```

will generate errors.

Note that once a numeric series or alpha series is defined to be auto-updating, its values may not be modified directly, since they are determined from the formula. Thus, if Z is an auto-updating series, the assignment command:

```
z = log(x)
```

will generate an error since an auto-updating series may not be modified. To modify Z you must either issue a new `frml` assignment or you must first set the values of Z to their current values by turning off auto-updating, and then issue the assignment statement.

To reset the formula in Z, you may simply issue the command:

```
frml z = log(x)
```

to replace the formula currently in the series.

To turn off auto-updating for a series, you should use the special expression "@CLEAR" in your `frml` assignment. When you turn off auto-updating, EViews freezes the numbers or strings in the series at their current values. Once the series is set to current values, it is treated as an ordinary series, and may be modified as desired. Thus, the commands:

```
frml z = @clear
z = log(x)
```

are allowed since Z is converted into an ordinary series prior to performing the series assignment.

One particularly useful feature of auto-updating series is the ability to reference series in databases. The command:

```
frml gdp = usdata::gdp
```

creates a series in the workfile called GDP that gets its values from the series GDP in the database USDATA. Similarly:

```
frml lgdp = log(usdata::gdp)
```

creates an auto-updating series named LGDP that contains the log of the values of GDP in the database USDATA.

Series that reference data in databases are *refreshed each time a workfile is loaded from disk*. Thus, it is possible to setup a workfile so that its data are current relative to a shared database.

Naming an Auto-Series

If you have previously opened a window containing an ordinary auto-series, you may convert the auto-series into an auto-updating series by assigning a name. To turn an auto-series into an auto-updating series, simply click on the **Name** button on the toolbar, or select **Object/Name...** from the main menu, and enter a name. EViews will assign the name to the series object, and will apply the auto-series definition as the formula to use for auto-updating.

Suppose, for example, that you have opened a series window containing an auto-series for the logarithm of the series CP by clicking on the **Show** button on the toolbar, or selecting **Quick/Show…** and entering "LOG(CP)". Then, simply click on the **Name** button in the auto-series toolbar, and assign a name to the temporary object to create an auto-updating series in the workfile.

Additional Issues

Auto-updating series are designed to calculate their values when in use, and automatically update values whenever the underlying data change. An auto-updating series will assign a value to every observation in the current workfile, irrespective of the current values of the workfile sample.

In most cases, there is no ambiguity in this operation. For example, if we have an auto-updating series containing the expression "LOG(CP)", we simply take each observation on CP in the workfile, evaluate the log of the value, and use this as the corresponding auto-updating series value.

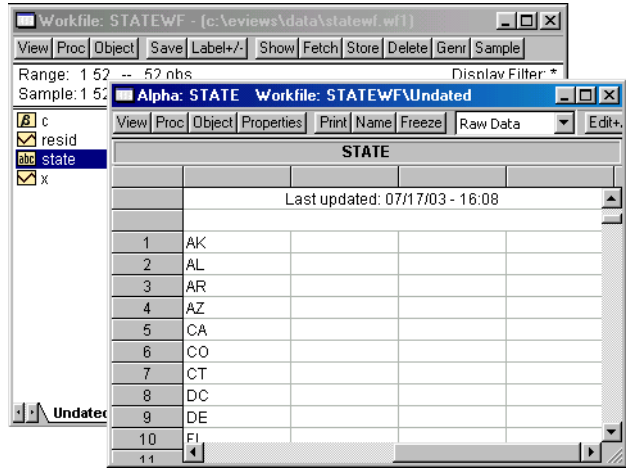
However, in cases where the auto-updating series contains an expression involving descriptive statistics, there is ambiguity as to whether the sample used to calculate the values is the sample at the time the auto-updating series was created, the sample at the time the series is evaluated, the entire workfile range, or some other sample.

To resolve this ambiguity, EViews will enter the current workfile sample into the expression at the time the auto-updating series is defined. Thus, if you enter "@MEAN(CP)" as your auto-updating series expression, EViews will substitute an expression of the form "@MEAN(CP, *smpl*)" into the definition. If you wish to evaluate the descriptive statistics for a given sample, you should enter an explicit sample in your expression.

Alpha Series

An *alpha series* object contains a set of observations on alphanumeric string values. Alpha series should be used when you wish to work with variables that contain alphanumeric data, such as names, addresses, and other text. If any of these types of data were entered into an ordinary series, EViews will replace the string with the numeric missing value, NA.

You may, for example, have an alpha series that contains the two-character U.S. Postal Service abbreviations for the 50 states, D.C., and Puerto Rico. Here, we show the alpha series, STATE, that contains the appropriate 2-character string values. STATE will be identified in the workfile with the alpha series icon labeled "abc", and by the designation **Alpha** in the titlebar of the alpha series window.



Similarly, alpha series may be used to hold identifying information such as the names and addresses of individuals, social security and telephone numbers, or classifying labels such as "male" and "female", or "high", "medium", and "low".

Declaring an Alpha Series

To create a new alpha series, you may select **Object/New Object...** from the main EViews window or workfile button bar, and then click on **Series Alpha** and optionally enter a name to be given to your alpha series. If you provide a name, EViews will create a new alpha series object in the workfile. If you do not supply a name, EViews will open an UNTITLED alpha series window.

Alternatively, you may type the keyword "ALPHA", followed by an optional series name, in the command window. The command:

```
alpha
```

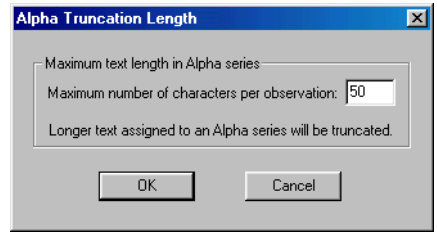
will create a new untitled alpha series and will display the series in an object window. Likewise:

```
alpha myseries
```

will create a new alpha series MYSERIES. To open the alpha series windows for MYSERIES or SETSERIES, simply double-click on the corresponding alpha series icon in the workfile window directory, or enter the command "SHOW MYSERIES".

In both of the cases described above, the alpha series will be initialized to contain missing values. For alpha series, the empty string (the null string, "") is used to designate a missing value. If you are declaring an alpha series using a command, you may combine the decla-

ration with the assignment of the values in the series. We explore alpha series assignment in "Assigning values to Alpha Series" on page 156.


Alpha Truncation Length
Maximum text length in Alpha series
Maximum number of characters per observation:
50
Longer text assigned to an Alpha series will be truncated.
OK
Cancel

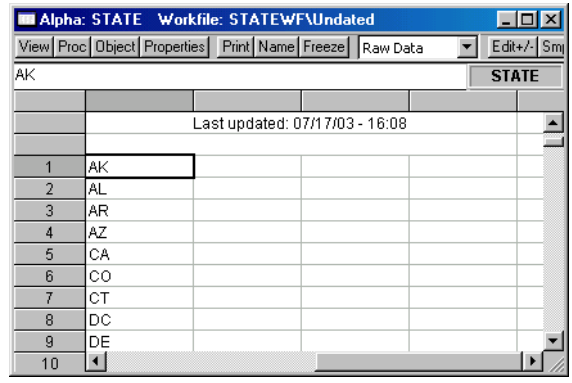
For the most part, you need not worry about the lengths of string values in your alpha series since EViews will automatically resize your series as required, up to the limit specified in the global defaults. Beyond that point, EViews will truncate the values of the alpha series. To modify the truncation length, select **Options/ Alpha Truncation...** from the main menu, and enter the desired length. Subsequent alpha series creation and assignment will use the new truncation length.

You should bear in mind that the strings in EViews alpha series are of fixed length so that the size of each observation is equal to the length of the longest string. If you have a series with all short strings with the exception of one very long string, the memory taken up by the series will be the number of observations times the longest string. In settings of this sort, efficiency suggests that you consider using value maps ("Value Maps" on page 163) to encode the values of the long string.

Editing an Alpha Series

There is no difference between editing an ordinary numeric series and editing an alpha series. Make certain that the alpha series is in edit mode by verifying the existence of the edit field in the series window. If not, click on the **Edit +/–** button to enable edit mode.

To edit a specific value, click on the desired cell. The existing value in the cell will appear in the edit window for you to modify or delete.



Simply type the new value in the edit window. Once you have entered the desired value, move to a new cell by clicking or using the arrow keys, or press the return key. This action will accept the entered value and prepare you for editing the newly selected cell.

Note that when editing the values of an alpha series, EViews does not require you to delimit your strings. You may simply type the relevant value in the edit field. EViews will remove any leading and trailing spaces from the value that you enter; if you wish to retain

those characters, enclose your string in double quotes. To enter the double quote character as a part of your string, you should escape the character with another double quote so that you enter two consecutive double quotes.

Assigning values to Alpha Series

You may assign values to an alpha series using string expressions. An alpha series assignment has the form:

alpha_name = string_expr

where *alpha_name* is the name of an existing alpha series and *string_expr* is any expression containing a combination of string literals, alpha series, and functions or operators that return strings (see "Strings" on page 119 of the *Command and Programming Reference* for details). As with ordinary series, we may combine the declaration and assignment steps so that the commands:

`alpha` *alpha_name = string_expr*

or

`genr` *alpha_name = string_expr*

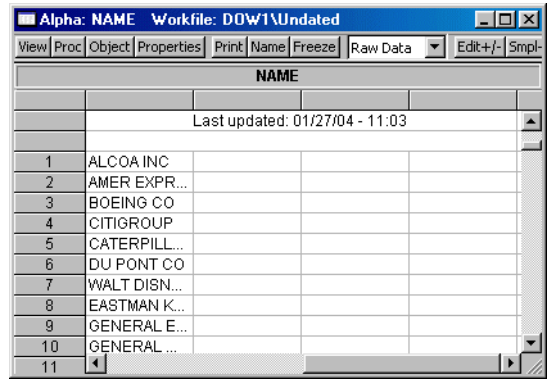
first create the alpha series *alpha_name* and then will assign the values using *string_expr*. In the latter command, EViews notes that the right-hand side expression is a string so that it knows to create an alpha series.

Alternatively, assuming that the alpha series exists, you may reassign the series values by clicking on **Quick/Generate Series...** in the main menu and entering the assignment and sample statements in the dialog. For example, if you enter the expression:

`myalpha` = *string_expr*

in the dialog, EViews will assign the values of the *string_expr* to the existing alpha series MYALPHA. Alternatively, you may enter the expression in the command line. In both cases, EViews will assign the corresponding values for all observations in the current workfile sample, overwriting the existing values.

Let us consider a simple example. Suppose that we have data on the company name (NAME), ticker symbol (SYMBOL), time of last trade (LAST_TIME), and closing price (LAST_TRADE) for each of the stocks in the Dow Jones Industrial Average on September 10, 2003.



Clicking on the icon for NAME, we can display the alpha series spreadsheet view. Note here that the default column width is not wide enough to display the contents of every observation, a condition that is signaled by the trailing "..." in the display for several of the observations. We may increase the column width by dragging the column header separators (the lines separating the column headers located just below the name of the series), by clicking on the **Properties** button and entering a larger number in the width field, or by double clicking on the column header separator to adjust the column width to the minimum width that displays all of the observation values without truncation.

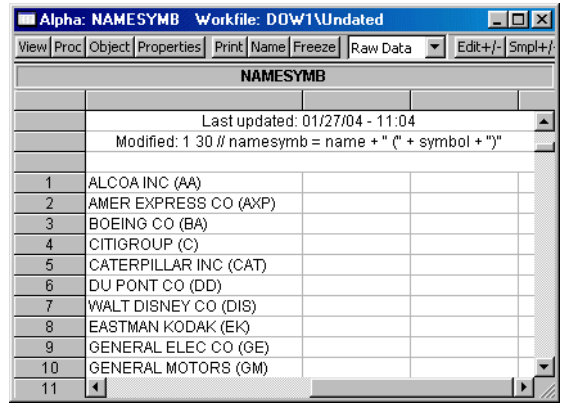
Suppose now that we wish to create an alpha series containing the name of each company followed by its ticker symbol (enclosed in parentheses). A simple assignment statement generates the desired series:

```
alpha namesymb = name + " (" + symbol + ")"
```

EViews will create a new alpha series NAMESYMB if one doesn't exist. Then, for every observation in the workfile sample, the contents of the alpha series NAME are concatenated with the literal strings for the parentheses, and the contents of the SYMBOL series.

Working with Alpha Series

Once created, an alpha series is used in two primary ways: (1) to generate numeric values and (2) to provide identifiers for the observations in the workfile.


Alpha: NAMESYMB Workfile: DOW1\Undated
View Proc Object Properties Print Name Freeze Raw Data Edit+/- Smpl+/-
NAMESYMB
Last updated: 01/27/04 - 11:04
Modified: 1 30 // namesymb = name + " (" + symbol + ")"
1 ALCOA INC (AA)
2 AMER EXPRESS CO (AXP)
3 BOEING CO (BA)
4 CITIGROUP (C)
5 CATERPILLAR INC (CAT)
6 DU PONT CO (DD)
7 WALT DISNEY CO (DIS)
8 EASTMAN KODAK (EK)
9 GENERAL ELEC CO (GE)
10 GENERAL MOTORS (GM)
11

Generating Numeric Values

By definition, an alpha series contains a string value for each observation. This means that if you use

an alpha series in a setting requiring numeric input, all of the values of the alpha series will be treated as NAs. For example, if you attempt to compute the mean of the STATE alpha series or use the Dow company NAME in an equation regression specification, EViews will generate an error saying that there are an insufficient number of observations, since all of the numeric values are missing.

You may, however, use the string relational operators (see "String Relational Operators" on page 121 of the *Command and Programming Reference*) to generate a series of numeric values. For the data from our Dow Jones example, the commands:

```
smpl @all
series wname = (@lower(@left(NAME, 1)) = "w")
```

generate the numeric series WNAME containing the value 1 if the company name begins with the letter "W", and 0 otherwise.

Similarly, the relational operators may be used when specifying a subsample. The command:

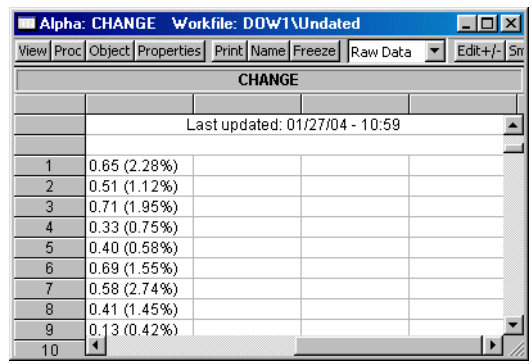
```
smpl @all if gender = "Male"
```

will restrict the workfile sample to include only observations where the string value in the alpha series GENDER is "Male".

You may also use the various functions described in "String Information Functions" on page 124 of the *Command and Programming Reference* to generate numeric values.

Two examples are of particular importance. First, you may have an alpha series that contains string representations of numbers, such as "3.14159". In order to use the strings as actual numbers, you must translate them into numbers, using either the string evaluation function @VAL.

Suppose, in our Dow Jones example, that we have the alpha series CHANGE containing information on the stock price change, expressed in both levels and percentages.

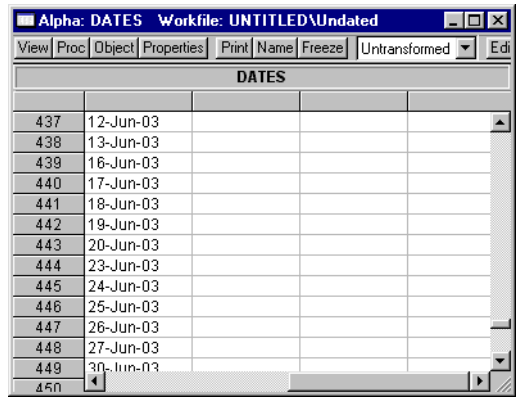


If we wish to extract only the levels information from the alpha series, the @LEFT function may be used to extract the leftmost four characters of each string. The @VAL function may then be used to obtain the numeric value for each observation. Putting this together, the command:

```
series chgval = @val(@left(change, 4))
```

converts the leading four characters of the CHANGE series into numeric values, and places the results in the series CHGVAL.

Second, you may have an alpha series that contains a text representation of dates. Here, we have a series DATES that contains text representations of dates in "dd-Mon-YY" format (one or two-digit day, dash, three-character month abbreviation, dash, two-digit year). For example, "12-Jun-03" represents June 12, 2003.



To convert every element of this series into a numeric series containing date values, simply issue the command:

```
series dval = @dateval(dates)
```

The newly created series DVAL will contain date numbers associated with each of the string values in DATES.

Additional Issues

The Spreadsheet View

By default, the alpha series spreadsheet will display your data left-justified, with a column width of approximately 12 characters. You may change the justification and column width settings by clicking on the **Properties** button in the toolbar, then selecting a new justification setting and entering a new column width. Alternatively, the column width may be changed by dragging the separator in the column header to the desired position, or by double-clicking on the separator to adjust the column width to the minimum width that displays all of the observation values without truncation.

Auto-series

You should note that like ordinary series, you may also work directly with a series expression that produces an alpha series. For example, if ALPHA1 is an alpha series, the command:

```
show @lower(alpha1)
```

will result in an alpha series containing the contents of ALPHA1 with the text converted to all lowercase characters.

Auto-series expressions involving alpha series may also evaluate to ordinary series. For example, if NUMER1 is a numeric series and ALPHA1 is an alpha series, you may enter:

```
show numer1+@len(alpha1)+(alpha1>"cat")
```

to open a series window containing the results of the operation. Note that the parentheses around the relational comparison are required for correct parsing of the expression.

Date Series

A *date series* is a standard EViews numeric series that contains valid date values (see "Dates" on page 129 of the *Command and Programming Reference*). There is nothing that distinguishes a date series from any other numeric series, except for the fact that the values it contains may be interpreted as dates.

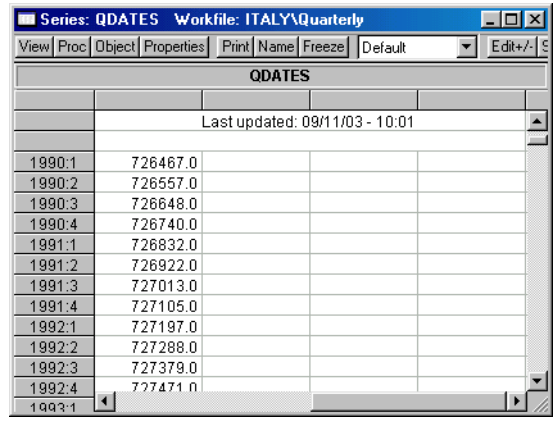
Creating a Date Series

There is nothing special about creating a date series. Any method of creating an EViews series may be used to create a numeric series that will be used as a date series.

Displaying a Date Series

The numeric values in a date series are generally of interest only when performing calendar operations. For most purposes, you will wish to see the values of your date series as date strings.

For example, the following series QDATES in our quarterly workfile is a numeric series containing valid date values for the start of each quarter. The numeric values of QDATES depicted here show the number of days since 1 January A.D. 1.



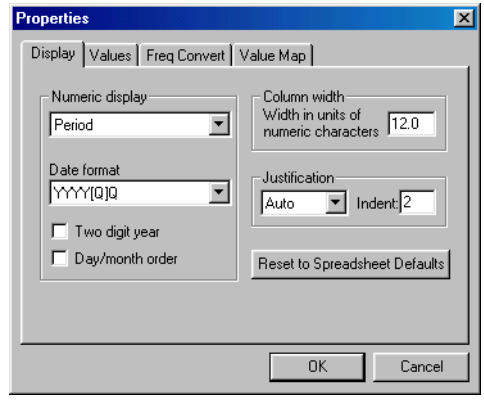
Obviously, this is not the way that most people will wish to view their date series. Accordingly, EViews provides considerable control over the display of the date values in your series. To change the display, click on the **Properties** button in the series toolbar, or select **View/Properties...** from the main menu.

EViews will display a dialog prompting you to change the display properties of the series. While you may change a variety of display settings such as the column width, justification,

and indentation, here, for the moment, we are more interested in setting the properties of the **Numeric Display**.

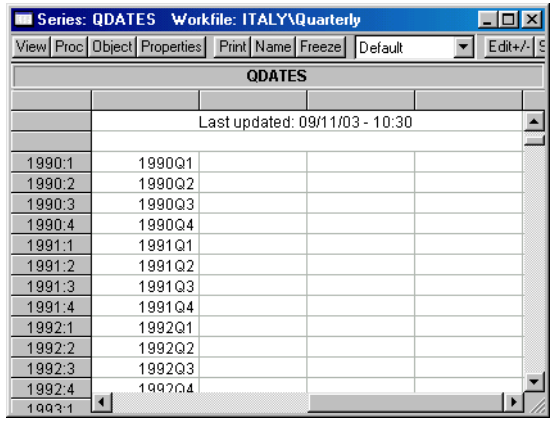
For a date series, there are four settings of interest in the **Numeric Display** combo box (**Period**, **Day**, **Day-Time**, and **Time**), with each corresponding to specific information that we wish to display in the series spreadsheet. For example, the **Day** selection allows you to display date information up to the day in various formats; with year, month, and day all included in the representation.

Let us consider our quarterly workfile example. Here we have selected **Period** and chosen a specific **Date format** entry ("YYYY[Q]Q"), which tells EViews that you wish to display the year, followed by the "Q" separator, and then the quarter number. Note also that when **Period** is selected, there is a **Current Workfile** setting in the **Date format** combo box which tells EViews to use the current workfile display settings.

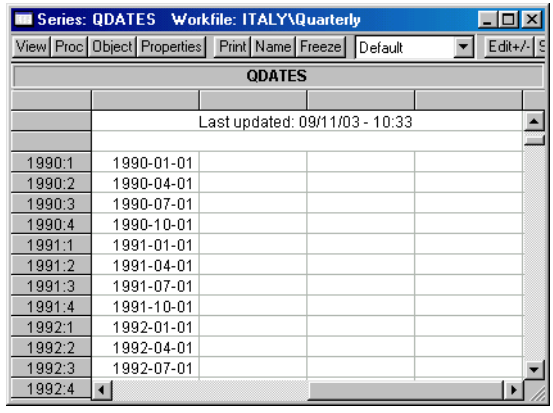


The two checkboxes below the **Date format** combo box may be used to modify the selected date format. If you select **Two digit year**, EViews will only display the last two-digits of the year (if the selected format is "YYYY[Q]Q", the actual format used will be "YY[Q]Q"); if you select **Day/month** order, days will precede months in whichever format is selected (if you select "mm/dd/YYYY" as the format for a day display, EViews will use "dd/mm/YYYY").

Applying this format to the QDATES series, the display changes to show the data in the new format:



If instead we select the **Day** display, and choose the "YYYY-mm-dd" format, the QDATES spreadsheet will show:

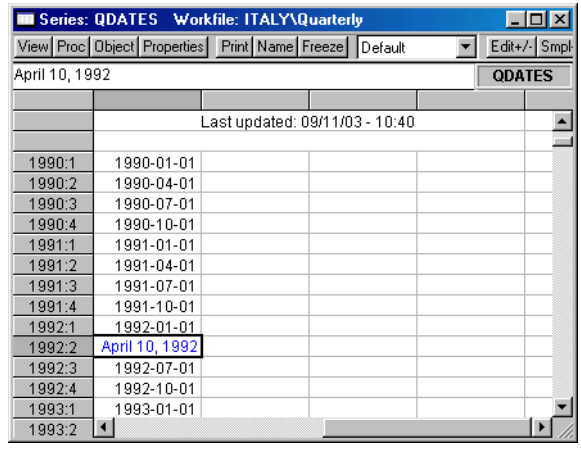


There is one essential fact to remember about the QDATES series. Despite the fact that we have changed the display to show a text representation of the date, QDATES still contains the underlying numeric date values. This is in contrast to using an alpha series to hold a text representation of the date.

If you wish to convert a (numeric) date series into an alpha series, you will need to use the @DATESTR function. If you wish to convert an alpha series into a numeric date series, you will need to use the @DATEVAL function. See "Translating between Date Strings and Date Numbers" on page 135 of the *Command and Programming Reference* for details.

Editing a Date Series

You may edit a date series either by either date numbers, or if the series is displayed using a date format, by entering date strings directly.



Suppose, for example, that we have our date series from above and that we wish to change a value. If we are displaying the series with date formatting, we may enter a date string, which EViews will automatically convert into a date number.

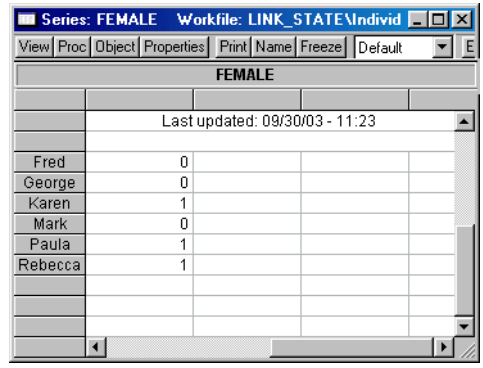
For example, we may edit our QDATES series by entering a valid date string ("April 10, 1992"), which EViews will convert into a date number (727297.0), and then display as a date string ("1992-04-10"). See "Free-format Conversion" on page 137 of the *Command and Programming Reference* for details on automatic translation of strings to date values.

Note, however, that if we were to enter the same string value in the series when the display is set to show numeric values, EViews will not attempt to interpret the string and will enter an NA in the series.

Value Maps

You may use the valmap object to create a *value map* (or *map*, for short) that assigns descriptive labels to values in numeric or alpha series. The practice of mapping observation values to labels allows you to hold your data in encoded or abbreviated form, while displaying the results using easy-to-interpret labels.

Perhaps the most common example of encoded data occurs when you have categorical identifiers that are given by integer values. For example, we may have a numeric series FEMALE containing a binary indicator for whether an individual is a female (1) or a male (0).



Numeric encodings of this type are commonly employed, since they allow one to work with the numeric values of FEMALE. One may, for example, compute the mean value of the FEMALE variable, which will provide the proportion of observations that are female.

On the other hand, numeric encoding of categorical data has the disadvantage that one must always translate from the numeric values to the underlying categorical data types. For example, a one-way tabulation of the FEMALE data produces the output:

Tabulation of FEMALE
Date: 09/30/03 Time: 11:36
Sample: 1 6
Included observations: 6
Number of categories: 2

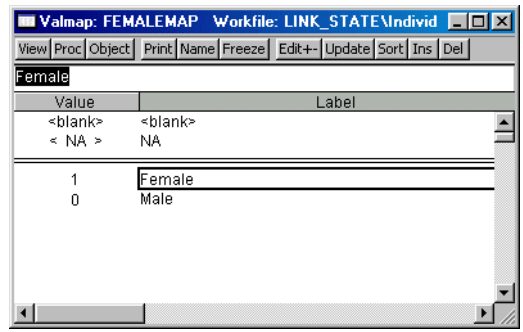
Value	Count	Percent	Cumulative Count	Cumulative Percent
0	3	50.00	3	50.00
1	3	50.00	6	100.00
Total	6	100.00	6	100.00

Interpretation of this output requires that the viewer remember that the FEMALE series is encoded so that the “0” value represents “Male” and the “1” represents “Female”. The example above would be easier to interpret if the first column showed the text representations of the categories in place of the numeric values.

Valmaps allow us to combine the benefits of descriptive text data values with the ease of working with a numeric encoding of the data. In cases where we define value maps for alphanumeric data, the associations allow us to use space saving abbreviations for the underlying data along with more descriptive labels to be used in presenting output.

Defining a Valmap

To create a valmap object, select **Object/New Object.../ValMap** from the main menu and optionally enter an object name, or enter the keyword "VALMAP" in the command line, followed by an optional name. EViews will open a valmap object.

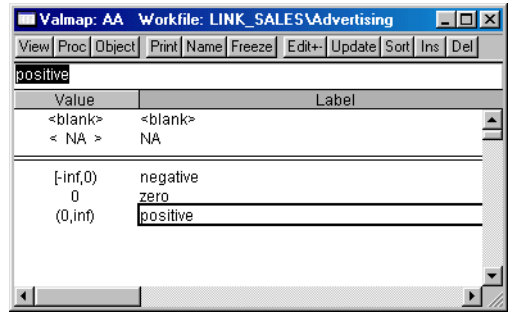


You will enter your new mappings below the double line by typing or by copy-and-pasting. In the **Value** column, you should enter the values for which you wish to provide labels; in the **Label** column, you will enter the corresponding text label. Here, we define a valmap named FEMALEMAP in which the value 0 is mapped to the string "Male", and the value 1 is mapped to the string "Female".

The two special entries above the double line should be used to define mappings for blank strings and numeric missing values. The default mapping is to represent blank strings as themselves, and to represent numeric missing values with the text "NA". You may change these defaults by entering the appropriate text in the **Label** column. For example, to change the representation of missing numeric values to, say, a period, simply type the "." character in the appropriate cell.

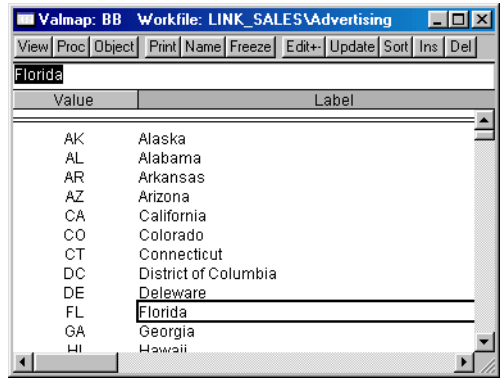
We caution that when working with maps, EViews will look for exact equality between the value in a series and the value in the valmap. Such an equality comparison is subject to the usual issues associated with comparing floating point numbers. To mitigate these issues and to facilitate mapping large numbers of values, EViews allows you to define value maps using intervals.

To map an interval, simply enter a range of values in the **Value** column and the associated label in the **Label** column. You may use round and square parentheses, to denote open ("(", ")") or closed ("[", "]") interval endpoints, and the special values "–INF" and "INF" to represent minus and plus infinity.



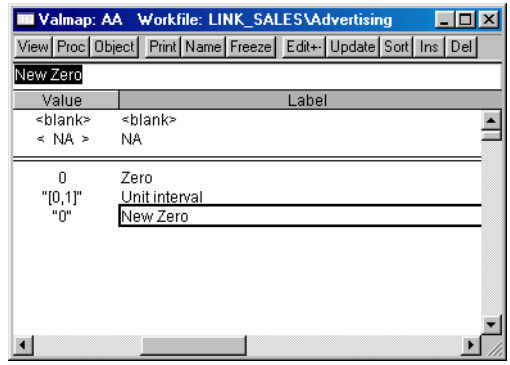
Using interval mapping, we require only three entries to map all of the negative values to the string "negative", the positive values to the string "positive", and the value 0 to the string "zero". Note that the first interval in

our example, "[–inf, 0)", is mathematically incorrect since the lower bound is not closed, but EViews allows the closed interval syntax in this case since there is no possibility of confusion.



While the primary use for valmaps will be to map numeric series values, there is nothing stopping you from defining labels corresponding to alpha series values (note that value and label matching is case sensitive). One important application of string value mapping is to expand abbreviations. For example, one might wish to map the U.S. Postal Service state abbreviations to full state names.

Since valmaps may be used with both numeric and alpha series, the text entries in the **Value** column may generally be used to match both numeric and alphanumeric values. For example, if you enter the text "0" as your value, EViews treats the entry as representing either a numeric 0 or the string value "0". Similarly, entering the text string "[0,1]" will match both numeric values in the interval, as well as the string value "[0,1]".



There is one exception to this dual interpretation. You may, in the process of defining a given valmap, provide an entry that conflicts with a previous entry. EViews will automatically identify the conflict, and will convert the latter entry into a string-only valmap entry.

For example, if the first line of your valmap maps 0 to "Zero", a line that maps 0 to "New Zero", or one that maps "[0, 1]" to "Unit interval" conflicts with the existing entry. In the latter cases, the conflicting maps will be treated as text maps. Such a map is identified by enclosing the entry with quotation marks. Here, EViews has automatically added the enclosing quotation marks to indicate that the latter two label entries will only be interpreted as string maps, and not as numeric maps.

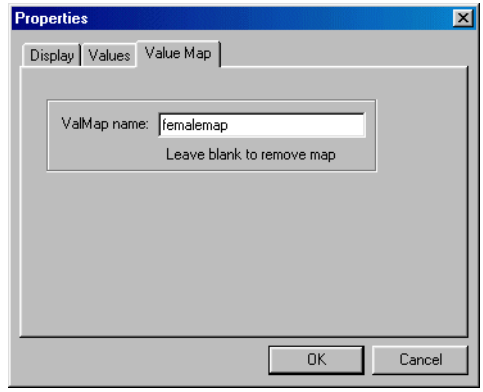
Once you have defined your mappings, click on the **Update** button on the toolbar to validate the object. EViews will examine your valmap and will remove entries with values that

are exact duplicates. In this example, the last entry, which maps the string "0" to the value "New Zero" will be removed since it conflicts with the first line. The second entry will be retained since it is not an exact duplicate of any other entry. It will, however, be interpreted only as a string since the numeric interpretation would lead to multiple mappings for the value 0.

Assigning a Valmap to a Series

To use a valmap, you need to instruct EViews to display the values of the map in place of the underlying data. Before working with a valmap, you should be certain that you have updated and validated your valmap by pressing the **Update** button on the valmap toolbar.

First, you must assign the value map to your series by modifying the series properties. Open the series window and select **View/Properties...** or click on the **Properties** button in the series toolbar to open the properties dialog. Click on the **Value Map** tab to display the value map name edit field.



If the edit field is blank, a value map has not been associated with this series. To assign a valmap, simply enter the name of a valmap object in the edit field and click on **OK**.

EViews will validate the entry and apply the specified map to the series. Note that to be valid, the valmap must exist in the same workfile page as the series to which it is assigned.

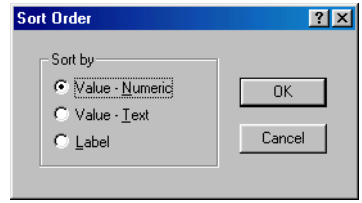
Using the Valmap Tools

EViews provides a small set of object-specific views and procedures that will aid you in working with valmaps.

Sorting Valmap Entries

You may add your valmap entries in any order without changing the behavior of the map. However, when viewing the contents of the map, you may find it useful to see the entries in sorted order.

To sort the contents of your map, click on **Proc/Sort...** from the main valmap menu. EViews provides you with the choice of sorting by the value column using numeric order (**Value - Numeric**), sorting by the value column using text order (**Value - Text**) or sorting by the label column (**Label**).



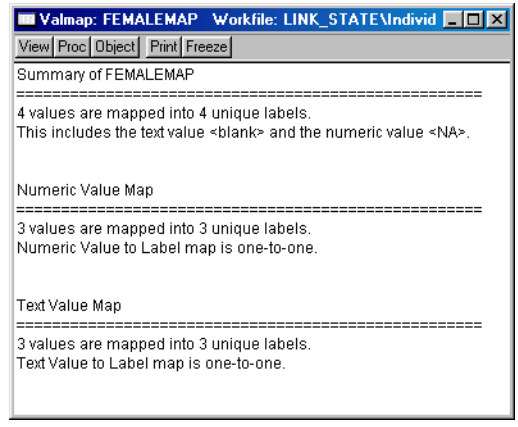
In the first two cases, we sort by the values in the first column of the valmap. The difference between the choices is apparent when you note that the ordering of the entries "9" and "10" depends upon whether we are interpreting the sort as a numeric sort, or as a text sort. Selecting **Value - Numeric** tells EViews that where possible, you wish to interpret strings as numbers when performing comparisons (so that "9" is less than "10"); selecting **Value - Text** says that all values should be treated as text for purposes of comparison (so that "10" is less than "9").

Click on **OK** to accept the sort settings.

Examining Properties of a Valmap

You may examine a summary of your valmap by selecting **View/Statistics** in the valmap window. EViews will display a view showing the properties of the labels defined in the object.

The top portion of the view shows the number of mappings in the valmap, and the number of unique labels used in those definitions. Here we see that the valmap has four definitions, which map four values into four unique labels. Two of the four definitions are the special entries for blank strings and the numeric NA value.


Valmap: FEMALEMAP Workfile: LINK_STATE\Individ
View
Proc
Object
Print
Freeze
Summary of FEMALEMAP
4 values are mapped into 4 unique labels.
This includes the text value <blank> and the numeric value <NA>.
Numeric Value Map
3 values are mapped into 3 unique labels.
Numeric Value to Label map is one-to-one.
Text Value Map
3 values are mapped into 3 unique labels.
Text Value to Label map is one-to-one.

The remaining portions of the view provide a detailed summary of the valmap describing the properties of the map when applied to numeric and to text values.

When applied to an ordinary numeric series, our FEMALEMAP example contains three relevant definitions that provide labels for the values 0, 1, and NA. Here, EViews reports that the numeric value mapping is one-to-one since there are no two values that produce the same value label.

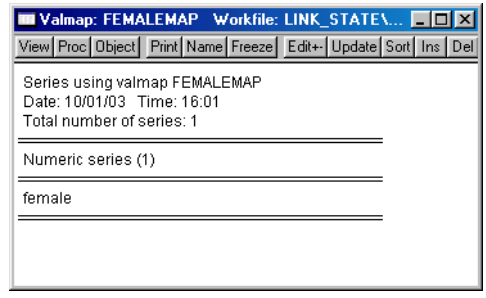
The output also reports that the FEMALEMAP has three relevant definitions for mapping the three text values, "0", "1", and the blank string, into three unique labels. We see that the text interpreted maps are also one-to-one.

Note that in settings where we map an interval into a given label, or where a given text label is repeated for multiple values, EViews will report a many-to-one mapping. Knowing that a valmap is many-to-one is important since it implies that the values of the underlying source series are not uniquely identified by the label values. This lack of identification has important implications in editing mapped series and in interpreting the results from various statistical output (see "Editing a Mapped Series" on page 171 and "Valmap Definition Cautions" on page 175).

Tracking Valmap Usage

A single valmap may be applied to more than one series. You may track the usage of a given valmap by selecting **View/Usage** from the valmap main menu. EViews will examine every numeric and alpha series in the workfile page to determine which, if any, have applied the specified valmap.

The valmap view then changes to show the number and names of the series that employ the valmap, with separate lists for the numeric and the alpha series. Here we see that there is a single numeric series named FEMALE that uses FEMALEMAP.



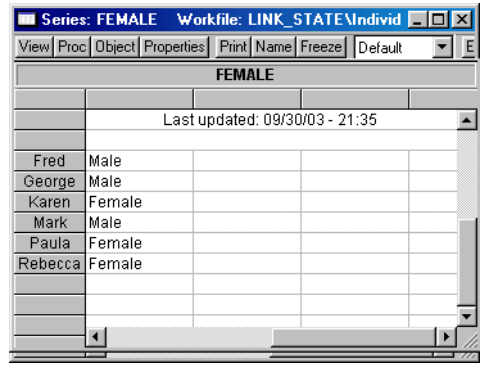
Working with a Mapped Series

Once you assign a map to a series, EViews allows you to display and edit your series using the mapped values and will use the labels when displaying the output from selected procedures.

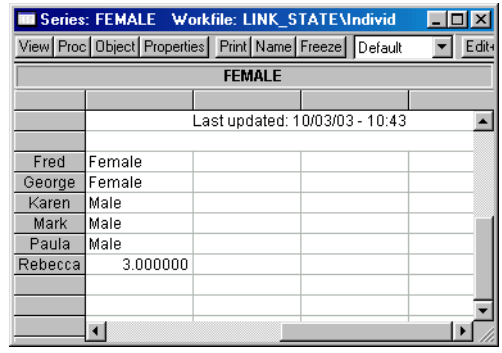
Displaying series values

By default, once you apply a value map to a series, the EViews spreadsheet view will change to display the newly mapped values.

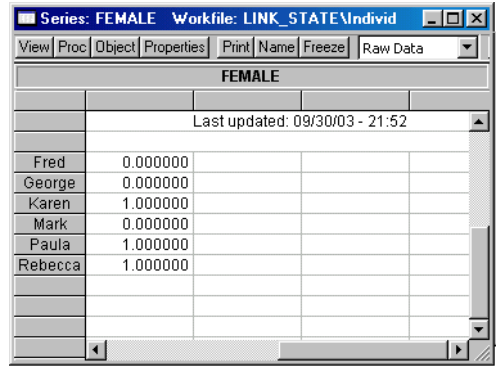
For example, after applying the FEMALE-MAP to our FEMALE series, the series spreadsheet view changes to show the labels associated with each value instead of the underlying encoded values. Note that the display format combo box usually visible in series toolbar indicates that EViews is displaying the **Default** series values, so that it shows the labels "Male" and "Female" rather than the underlying 0 and 1 values.



Note that if any of the values in the series does not have a corresponding valmap entry, EViews will display a mix of labeled and unlabeled values, with the unlabeled value "showing through" the mapping. For example, if the last observation in the FEMALE series had the value 3, the series spreadsheet will show observations with "Male" and "Female" corresponding to the mapped values, as well as the unmapped value 3.



There may be times when you wish to view the underlying series values instead of the labels. There are two possible approaches. First, you may remove the valmap assignment from the series. Simply go to the **Properties** dialog, and delete the name of the valmap object from the **Value Map** page. The display will revert to showing the underlying values. Less drastically, you may use the display method combo box to change the display format for the spreadsheet view. If you select **Raw Data**, the series spreadsheet view will change to show the underlying series data.



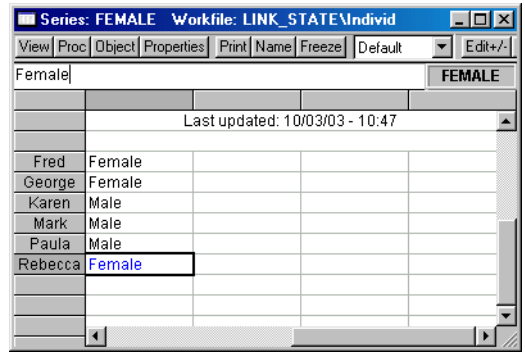
Editing a Mapped Series

To edit the values of your mapped series, first make certain you are in edit mode, then enter the desired values, either by typing in the edit field, or by pasting from the clipboard. How EViews interprets your input will differ depending upon the current display format for the series.

If your mapped series is displayed in its original form using the **Raw Data** setting, EViews will interpret any input as representing the underlying series values, and will place the input directly into the series. For example, if our FEMALE series is displayed using the **Raw Data** setting, any numeric input will be entered directly in the series, and any string input will be interpreted as an NA value.

In contrast, if the series is displayed using the **Default** setting, EViews will use the attached valmap both in displaying the labeled values and in interpreting any input. In this setting, EViews will first examine the attached valmap to determine whether the given input value is also a label in the valmap. If a matching entry is found, and the label matches a unique underlying value, EViews will put the value in the series. If there is no matching valmap label entry, or if there is an entry but the corresponding value is ambiguous, EViews will put the input value directly into the series. One implication of this behavior is that so long as the underlying values are not themselves valmap labels, you may enter data in either mapped or unmapped form. Note, again, that text value and label matching is case-sensitive.

Let us consider a simple example.



Suppose that the FEMALE series is set to display mapped values, and that you enter the value "Female". EViews will examine the assigned valmap, determine that "Female" corresponds to the valmap value "1", and then assign this value to the series. Since "1" is a valid form of numeric input, the value 1 will be placed in the series. Note that even though we have entered 1 into the series, the mapped spreadsheet view will continue to show the value "Female".

Alternatively, we could have entered a "1" corresponding to the underlying numeric value. Since "1" is not a valmap label, EViews will put the value 1 in the series, which will be displayed using the label "Female".

While quite useful, entering data in mapped display mode requires some care, as your results may be somewhat unexpected. For one, you should bear in mind that the required reverse lookup of values associated with a given input requires an exact match of the input to a label value, and a one-to-one correspondence between the given label and a valmap value. If this condition is not met, the original input value will be placed in the series.

Consider, for example, the result of entering the string "female" instead of "Female". In this case, there is no matching valmap label entry, so EViews will put the input value, "female", into the series. Since FEMALE is a numeric series, the resulting value will be an NA, and the display will show the mapped value for numeric missing values.

Similarly, suppose you enter "3" into the last observation of the FEMALE series. Again, EViews does not find a corresponding valmap label entry, so the input is entered directly into the series. In this case, the input represents a valid number so that the resulting value will be a 3. Since there is no valmap entry for this value, the underlying value will be displayed.

Lastly, note that if the matching valmap label corresponds to multiple underlying values, EViews will be unable to perform the reverse lookup. If, for example, we modify our valmap so that the interval "[1, 10]" (instead of just the value 1) maps to the label "Female", then when you enter "Female" as your input, it is impossible to determine a unique value for the series. In this case, EViews will enter the original input, "Female", directly into the series, resulting in an NA value.

See "Valmap Definition Cautions" on page 175 for additional cautionary notes.

Using a Mapped Series

You may use a mapped series as though it were any series. We emphasize the fact that the mapped values of a series are not replacements of the underlying data; they are only labels to be used in output. Thus, when performing numeric calculations with a series, EViews will always use the underlying values of the series, not the label values. For example, if you map the numeric value -99 to the text "NA", and take the absolute value of the mapped numeric series containing that value, you will get the value 99, and not a missing value.

In appropriate settings (where the series values are treated as categories), EViews routines will use the labels when displaying output. For example, a one-way frequency tabulation of the FEMALE series with the assigned FEMALEMAP yields:

Tabulation of FEMALE
Date: 10/01/03 Time: 09:27
Sample: 1 6
Included observations: 6
Number of categories: 2

Value	Count	Percent	Cumulative Count	Cumulative Percent
Male	3	50.00	3	50.00
Female	3	50.00	6	100.00
Total	6	100.00	6	100.00

Similarly, when computing descriptive statistics for the SALES data categorized by the values of the FEMALE series, we have:

Descriptive Statistics for SALES
Categorized by values of FEMALE
Date: 10/01/03 Time: 09:30
Sample: 1 6
Included observations: 6

FEMALE	Mean	Std. Dev.	Obs.
Male	323.3333	166.2328	3
Female	263.3333	169.2139	3
All	293.3333	153.5795	6

Valmap Functions

To facilitate working with valmaps, three new `genr` functions are provided which allow you to translate between unmapped and mapped values. These functions may be used as part of standard series or alpha expressions.

First, to obtain the mapped values corresponding to a set of numbers or strings, you may use the command:

@MAP(*arg[, map_name]*)

where *arg* is a numeric or string series expression or literal, and the optional *map_name* is the name of a valmap object. If *map_name* is not provided, EViews will attempt to determine the map by inspecting *arg*. This attempt will succeed only if *arg* is a numeric series or alpha series that has previously been mapped.

Let us consider our original example where the FEMALEMAP maps 0 to "Male" and 1 to "Female". Suppose that we have two series that contain the values 0 and 1. The first series,

MAPPEDSER, has previously applied the FEMALEMAP, while the latter series, UNMAPPEDSER, has not.

Then the commands:

```
alpha s1 = @map(mappedser)
alpha s2 = @map(mappedser, femalemap)
```

are equivalent. Both return the labels associated with the numeric values in the series. The first command uses the assigned valmap to determine the mapped values, while the second uses FEMALEMAP explicitly.

Alternately, the command:

```
alpha s3 = @map(unmappedser)
```

will generate an error since there is no valmap assigned to the series. To use @MAP in this context, you must provide the name of a valmap, as in:

```
alpha s4 = @map(unmappedser, femalemap)
```

which will return the mapped values of UNMAPPEDSER, using the valmap FEMALEMAP.

Conversely, you may obtain the numeric values associated with a set of string value labels using the @UNMAP function. The @UNMAP function takes the general form:

@UNMAP(*arg, map_name*)

to return the numeric values that have been mapped into the string given in the string expression or literal *arg*, where *map_name* is the name of a valmap object. Note that if a given label is associated with multiple numeric values, the missing value NA will be returned. Note that the *map_name* argument is required with the @UNMAP function.

Suppose, for example, that you have an alpha series STATEAB that contains state abbreviations (“AK”, “AL”, etc.) and a valmap STATEMAP that maps numbers to the abbreviations. Then:

```
series statecode = @unmap(stateab, statemap)
```

will contain the numeric values associated with each value of STATEAB.

Similarly, you may obtain the string values associated with a set of string value labels using:

@UNMAPTXT(*arg, map_name*)

where *arg* is a string expression or literal, and *map_name* is the name of a valmap object. @UNMAPTXT will return the underlying string values that are mapped into the string labels provided in *arg*. If a given label is associated with multiple values, the missing blank string “” will be returned.

Valmap Definition Cautions

EViews allows you to define quite general value maps that may be used with both numeric and alpha series. While potentially useful, the generality comes with a cost, since if used carelessly, valmaps can cause confusion. Accordingly, we caution you that there are many features of valmaps that should be used with care. To illustrate the issues, we list a few of the more problematic cases.

Many-to-one Valmaps

A many-to-one valmap is a useful tool for creating labels that divide series values into broad categories. For example, you may assign the label "High" to a range of values, and the label "Low" to a different range of values so that you may, when displaying the series labels, easily view the classification of an observation.

The downside to many-to-one valmaps is that they make interpreting some types of output considerably more difficult. Suppose, for example, that we construct a valmap in which several values are mapped to the label "Female". If we then display a one-way frequency table for a series that uses the valmap, the label "Female" may appear as multiple entries. Such a table is almost impossible to interpret since there is no way to distinguish between the various "Female" values.

A series with an attached many-to-one valmap is also more difficult to edit when viewing labels since EViews may be unable to identify a unique value corresponding to a given label. In these cases, EViews will assign a missing value to the series, which may lead to confusion (see "Editing a Mapped Series" on page 171).

Mapping Label Values

Defining a map in which one of the label values is itself a value that is mapped to a label can cause confusion. Suppose, for example, that we have a valmap with two entries: the first maps the value 6 to the label "six", and the second maps the value "six" to the label "high".

Now consider editing an alpha series that has this valmap attached. If we use the **Default** display, EViews will show the labeled values. Thus, the underlying value "six" will display as the value "high"; while the value "6" will display as "six". Since the string "six" is used both as a label and as a value, in this setting we have the odd result that it must be entered indirectly. Thus, to enter the string "six" in the alpha series, we have the counterintuitive result that you must type "high" instead of "six", since entering the latter value will put "6" in the series.

Note, however, that if you display the series in **Raw Data** form, all data entry is direct; entering "six" will put the value "six" into the series and entering "high" will put the value "high" in the series.

Mapping Values to Numbers

Along the same lines, we strongly recommend that you not define value maps in which numeric values can be mapped to labels that appear to be numeric values. Electing, for example, to define a valmap where the value 5 is mapped to the label “6” and the value 6 is mapped to the label “5”, is bound to lead to confusion.

Chapter 8. Series Links

The *series link* object (or *link*, for short) provides you with powerful tools for combining information from different workfile pages. Links provide an easy-to-use interface to a wide range of sophisticated data operations such as:

- merging data from one workfile page into another
- saving "by-group" summary statistics into a workfile page
- matching observations between dated workfile pages
- performing frequency conversion between regular dated workfile pages

Links operate both dynamically and on demand, so that the desired operation is performed only when needed, and is updated automatically whenever your data change.

You may find that working with links is in many ways similar to working with data tables in a relational database management system. Indeed, links have specifically been designed to provide much of the power of these sophisticated systems. But you need not have worked with such a system to take advantage of the power, ease-of-use, and flexibility associated with link objects.

We begin with a discussion of basic link concepts that outlines the basic operations supported by links. In later sections we document the use of links in EViews.

Basic Link Concepts

A link is a series-like object that exists in one workfile page, but "refers" to series data in another workfile page. At a basic level, a link is a description of how EViews should use data in a *source* workfile page to determine values of a series in the current, or *destination,* workfile page.

A link contains three fundamental components:

- First, there is the name of a *source series.* The source series identifies the series in the source workfile page that is used as a basis for obtaining values in the destination page.
- Second, the link contains the names of one or more link *identifier (ID)* series in both the source and destination pages. The source ID and destination ID series will be used to match observations from the two pages.
- Lastly, the link contains a description of how the source series should be used to construct link values for matching observations in the destination page.

The basic series link employs a method called *match merging* to determine the link values in the destination page. More advanced links combine match merging with automatic frequency conversion. We describe these two methods in detail below, in "Linking by general match merging" on page 178 and "Linking by date with frequency conversion" on page 187.

As the name suggests, the series link object shares most of the properties of a series. You may, in fact, generally use a series link as though it were a series. You may examine series views, perform series procedures, or use the series link to generate new data, or you may use the link as a regressor in an equation specification.

Another important property of links is that they are "live", in the sense that the values in the link change as its underlying data change. Thus, if you have a link in a given workfile page, the link values will automatically be updated when the source series or ID series values change.

Lastly, links are memory efficient. Since links are computed and updated as needed, the values of the series link are not held in memory unless they are in use. Thus, it is possible to create a page populated entirely by links that takes up only the minimum amount of memory required to perform all necessary operations.

Linking by general match merging

We begin our discussion of linking with a brief, and admittedly terse, description of how a basic link with match merging works. More useful, perhaps, will be the extended examples that follow.

The basic link first compares values for one or more source ID series with the values in the destination ID series. Observations in the two pages are said to match if they have identical ID values. When matches are observed, values from the source series are used to construct values of the link for the corresponding observations in the destination page.

Each link contains a description of how the source series should be used to construct link values in the destination page. Constructing values for a basic match merge link involves two steps:

- First, we perform a *contraction* of the source series to ensure that there is a single value associated with each distinct source ID value. The contraction method employed describes how the (possibly) multiple source series observations sharing a given ID value should be translated into a single value.
- Next, we take the distinct source IDs and contracted source series values, and perform a match merge in which each contracted value is repeated for all matching observations in the destination page.

This basic method is designed to handle the most general cases involving many-to-many match merging by first computing a many-to-one contraction (by-group summary) of the source series, and then performing a one-to-many match merge of the contracted data.

All other match merges are handled as special cases of this general method. For a many-to-one match merge, we first compute the contraction, then perform one-to-one matching of the contracted data into the destination page. In the more common one-to-many or one-to-one match merge, the contraction step typically has no practical effect since the standard contractions simply return the original source series values. The original values are then linked into the destination page using a simple one-to-one or one-to-many match merge.

While all of this may seem a bit abstract, a few simple examples should help to fix ideas. Suppose first that we have a state workfile page containing four observations on the series STATE1 and TAXRATE:

State1	TaxRate
Arkansas	.030
California	.050
Texas	.035
Wyoming	.012

In the same workfile, we have a second workfile page containing individual level data, with a name, NAME, state of residence, STATE2, and SALES volume for six individuals:

Name	State2	Sales
George	Arkansas	300
Fred	California	500
Karen	Arkansas	220
Mark	Texas	170
Paula	Texas	120
Rebecca	California	450

We wish to link the data between the two pages. Note that in this example, we have given the state series different names in the two pages to distinguish between the two. In practice there is no reason for the names to differ, and in most cases, the names will be the same.

One-to-many match merge

Our first task will be to create, in the page containing individual information, a series containing values of the TAXRATE faced by every individual. We will determine the individual

rates by examining each individual's state of residence and locating the corresponding tax rate. George, for example, who lives in Arkansas, will face that state's tax rate of 0.030. Similarly, Mark, who lives in Texas, has a tax rate of 0.035.

We will use a series link to perform a one-to-many match merge in which we assign the TAXRATE values in our source page to multiple individuals in our destination page.

For the three basic components of this link, we define:

- the source series TAXRATE
- the source identifier STATE1 and destination identifier STATE2
- the merge rule that the values of TAXRATE will be repeated for every individual with a matching STATE2 value in the destination page

This latter merge rule is always used for basic links involving one-to-many match merges. Here, the rule leads to the natural result that each individual is assigned the TAXRATE value associated with his or her state.

After performing the link, the individual page will contain the merged values for the tax rate in TAXRATE2. We use the "2" in the TAXRATE2 name to denote the fact that these data are generated by merging data using STATE2 as the destination ID series:

Name	State2	Sales	TaxRate2
George	Arkansas	300	.030
Fred	California	500	.050
Karen	Arkansas	220	.030
Mark	Texas	170	.035
Paula	Texas	120	.035
Rebecca	California	450	.050

We mention one other issue in passing that will become relevant in later discussion. Recall that all basic links with match merging first contract the source series prior to performing the match merge. In this case, the specified merge rule implicitly defines a contraction of the source series TAXRATE that has no effect since it returns the original values of TAXRATE. It is possible, though generally not desirable, to define a contraction rule which will yield alternate source values in a one-to-many match merge. See "Link calculation settings" on page 193.

Many-to-one match merge

Alternatively, we may wish to link data in the opposite direction. We may, for example, choose to link the SALES data from the individual page to the destination state page, again

matching observations using the two state IDs. This operation is a many-to-one match merge, since there are many observations with STATE2 ID values in the individual page for each of the unique values of STATE1 in the state page.

The components of this new link are easily defined:

- the source series SALES
- the source identifier STATE2 and destination identifier STATE1
- a merge rule stating that the values of SALES will first be contracted, and that the contracted values will be placed in matching observations in the destination page

Specifying the last component, the merge rule, is a bit more involved here since there are an unlimited number of ways that we may contract the individual data. EViews provides an extensive menu of contraction methods. Obvious choices include computing the mean, variance, sum, minimum, maximum, or number of observations for each source ID value. It is worth noting here that only a subset of the contraction methods are available if the source is an alpha series.

To continue with our example, suppose that we choose to take the sum of observations as our contraction method. Then contraction involves computing the sum of the individual observations in each state; the summary value for SALES in Arkansas is 520, the value in California is 950, and the value in Texas is 290. Wyoming is not represented in the individual data, so the corresponding contracted value is NA.

Given this link definition, the many-to-one match merge will result in a state page containing the match merged summed values for SALES1:

State1	TaxRate	Sales1	Sales1ct
Arkansas	.030	520	2
California	.050	950	2
Texas	.035	290	2
Wyoming	.012	NA	0

Similarly, we may define a second link to the SALES data containing an alternative contraction method, say the count of non-missing observations in each state. The resulting link, SALES1CT, shows that there are two individual observations for each of the first three states, and none for Wyoming.

Many-to-many match merge

Lastly, suppose that we have a third workfile page containing a panel structure with state data observed over a two year period:

Year	State3	TaxRate
1990	Arkansas	.030
1991	Arkansas	.032
1990	California	.050
1991	California	.055
1990	Texas	.035
1991	Texas	.040
1990	Wyoming	.012
1991	Wyoming	.035

Linking the SALES data from the *individual* page to the panel page using the STATE2 and STATE3 identifiers involves a many-to-many match merge since there are multiple observations for each state in both pages.

The components of this new link are easily defined:

- the source series SALES
- the source identifier STATE2 and destination identifier STATE3
- a merge rule stating that the values of SALES will first be contracted, and that the contracted values will be repeated for every observation with a matching STATE3 value in the destination page

This merge rule states that we perform a many-to-many merge by first contracting the source series, and then performing a one-to-many match merge of the contracted results into the destination. For example, linking the SALES data from the individual page into the panel state-year page using the sum and count contraction methods yields the link series SALES3 and SALES3A:

Year	State3	TaxRate	Sales3	Sales3a
1990	Arkansas	.030	520	2
1991	Arkansas	.032	520	2
1990	California	.050	950	2
1991	California	.055	950	2
1990	Texas	.035	290	2
1991	Texas	.040	290	2
1990	Wyoming	.012	NA	0
1991	Wyoming	.035	NA	0

It is worth noting that this many-to-many match merge is equivalent to first performing a many-to-one link from the individual page into the state page, and then constructing a one-to-many link of those linked values into the panel page. This two-step method may be achieved by first performing the many-to-one link into the state page, and then performing a one-to-many link of the SALES1 and SALES1CT links into the panel page.

Linking by date match merging

To this point, we have primarily considered simple examples involving a single categorical link identifier series (states). You may, of course, construct more elaborate IDs using more than one series. For example, if you have data on multinational firms observed over time, both the firm and date identifiers may be used as the link ID series.

The latter example is of note since it points to the fact that dates may be used as valid link identifiers. The use of dates as identifiers requires special discussion, as the notion of a match may be extended to take account of the calendar.

We begin our discussion of merging using dates by noting that a date may be employed as an identifier in two distinct ways:

- First, an ID series containing date values or alphanumeric representations of dates may be treated like any other ID series. In this case, the value in one workfile page must be *identical* to the value in the other page for a match to exist.
- Alternatively, when we are working with regular frequency data, we may take advantage of our knowledge of the frequency and the calender to define a broader notion of date matching. This broader form of matching, which we term *date matching*, involves comparing dates by first rounding the date ID values down to the lowest common regular frequency and then comparing the rounded values. Note that date matching requires the presence of at least one regular frequency for the rounding procedure to be well-defined.

In practical terms, date matching produces the outcomes that one would naturally expect. With date matching, for example, the quarterly observation "2002Q1" matches "2002" in a regular annual workfile, since we round the quarterly observation down to the annual frequency, and then match the rounded values. Likewise, we would match the date "March 3, 2001" to the year 2001 in an annual workfile, and to "2001Q1" in a quarterly workfile. Similarly, the date "July 10, 2001" also matches 2001 in the annual workfile, but matches "2001Q3" in the quarterly workfile.

Basic links with date matching

Consider the following simple example of linking using date matching. Suppose that we have a workfile containing two pages. The first page is a regular frequency quarterly page containing profit data (PROFIT) for 2002 and 2003:

Quarter	Profit
2002Q1	120
2002Q2	130
2002Q3	150
2002Q4	105
2003Q1	100
2003Q2	125
2003Q3	200
2003Q4	170

while the second page contains irregular data on special advertising events (ADVERT):

Date	Advert
Jan 7, 2002	10
Mar 10, 2002	50
Apr 9, 2002	40
May 12, 2002	90
Mar 1, 2003	70
Dec 7, 2003	30
Dec 23, 2003	20

Using QUARTER as the source ID and DATE as the destination ID, we link the quarterly profit data to the advertising page. The quarterly values in the source page are unique so that we have a one-to-many match merge; accordingly, we may select any contraction method that leaves the original PROFIT data unchanged (mean, unique, *etc.*).

Employing date matching at the quarterly frequency, we construct a PROFIT1 link containing the values:

Date	Advert	Profit1
Jan 7, 2002	10	120
Mar 10, 2002	50	120
Apr 9, 2002	40	130
May 12, 2002	90	130
Mar 1, 2003	70	100
Dec 7, 2003	30	170

Dec 23, 2003	20	170

In evaluating the values in PROFIT1, we simply repeat the value of PROFIT for a given quarter for every matching observation in the advertising page. For example, the observation for quarter "2002Q1" matches both "Jan 7, 2002" and "Mar 10, 2002" in the advertising page so that the latter observations are assigned the value of 120.

Conversely, using date matching to link the ADVERT series to the quarterly page, we have a many-to-one match merge since, after rounding, multiple observations in the advertising page have ID values that match the unique ID values in the quarterly page. If we choose to employ the mean contraction method in the link ADVERT1, we have:

Quarter	Profit	Advert1
2002Q1	120	30
2002Q2	130	65
2002Q3	150	NA
2002Q4	105	NA
2003Q1	100	70
2003Q2	125	NA
2003Q3	200	NA
2003Q4	170	25

Here, the values of ADVERT1 contain the mean values over the observed days in the quarter. For example, the value for ADVERT1 in 2002Q1 is taken by averaging the values of ADVERT for "Jan 7, 2002" and "Mar 10, 2002". Note that the value for quarter 2002Q3 is NA since there are no observations with matching DATE values, *i.e.*, there are no observations in the advertising page that fall within the quarter.

Note that in both of these examples, had we employed exact matching using the values in QUARTER and DATE, we would have observed no matches. As a result, all of the values in the resulting links would be assigned the value NA.

Panel links with date matching

When using date matching to link dated panel data to a page with a different frequency, you should pay particular attention to the behavior of the merge operation since the results may differ from expectations.

An example will illustrate the issue. Consider the following simple panel featuring quarterly revenue data from 2002Q1 to 2003Q4:

Firm	Quarter	Revenue
1	2002Q1	120
1	2002Q2	130
1	2002Q3	150
1	2002Q4	105
1	2003Q1	100
1	2003Q2	125
1	2003Q3	200
1	2003Q4	170
2	2002Q1	40
2	2002Q2	40
2	2002Q3	50
2	2002Q4	35
2	2003Q1	20
2	2003Q2	25
2	2003Q3	50
2	2003Q4	40

We will consider the results from linking the REVENUE data into an annual page using date matching of the QUARTER and the YEAR identifiers. Using date match merging, and employing both the sum and number of observations contractions, we observe the results in REVENUE1 (sum) and REVENUE1A (obs):

Year	Revenue1	Revenue1a
2002	670	8
2003	730	8

The important thing to note here is that the sums for each year have been computed over all eight matching observations in the panel page.

The key to understanding the result is to bear in mind that date matching only changes the way that a match between observations in the two pages is defined; the remaining match merge operation remains unchanged. The outcome is simply the result of applying standard link behavior in which we first identify matches, compute a contraction over all matching observations, and perform the one-to-one match merge.

An alternative approach to obtaining annual revenue values from the panel data would be to first contract the panel data to a quarterly frequency by averaging across firms, and then

to convert the quarterly data to an annual frequency by summing over quarters. This approach produces very different results from the first method.

This alternative may be undertaken in two steps: by first linking the quarterly panel data into a quarterly page (using the mean contraction), and then frequency converting by linking the quarterly data to the annual frequency (using summing over quarters). See "Panel frequency conversion" on page 189 for additional discussion and a description of EViews tools for defining a single link that performs both steps.

Linking by date with frequency conversion

In the special case where we wish to link data between two regular frequency pages using dates as the sole identifier, EViews allows you to define your links in two ways. First, you may use the date match merging described in "Linking by date match merging" on page 183, or you can define special links that employ frequency conversion.

Basic frequency conversion

Links specified by date will primarily be used to perform automatic frequency conversion of simple regular frequency data. For example, you may choose to hold your quarterly frequency data in one page, your monthly frequency data in a second page, and to create links between pages which automatically perform the up or down frequency conversion as necessary.

You can instruct EViews to use the source series default methods for converting between frequencies, or you may use the link definition to specify the up and down conversion methods. Furthermore, the live nature of links means that changes in the source data will generate automatic updates of the frequency converted link values.

We divide our discussion of frequency conversion links into those that link data from high to low frequency pages and those that link from low to high frequency pages.

High to low frequency conversion

Frequency conversion linking from a simple regular high frequency page to a regular low frequency page is fundamentally the same as using a link with date matching to perform basic many-to-one match merging. In both cases, we match dates, compute a contraction of the source series, and then perform a one-to-one match merge.

Given the specialized nature of frequency conversion, links specified by date with frequency conversion offer a subset of the ordinary link contraction methods. All of the standard high to low frequency conversion methods (average, sum, first, last, maximum and minimum) are supported, but the match merge methods which do not preserve levels, (such as the sum-of-squares or the variance) are not included.

Frequency conversion links also allow you to disable conversions for partially observed periods, so that a missing value for the source series in a given month generates a missing value for the corresponding quarterly observation. This option is not available for basic match merge links.

Low to high- frequency conversion

In contrast, linking from low to high frequency pages using frequency conversion differs substantively from linking using basic date match merging.

When linking using general date match merging, the frequency conversion implied by the one-to-many match merge may only be performed by repeating the low frequency observation for every matching high frequency observation. Thus, in a one-to-many date match merge, an annual observation is always repeated for each matching quarter, month, or day.

In contrast, EViews provides additional up-conversion methods for frequency conversion links. In addition to the simple repeated-observation (constant-match average) method, frequency conversion links support all of the standard frequency conversion methods including constant-match sum, quadratic-match sum, quadratic-match average, linear-match sum, linear-match last, and cubic-match last.

Suppose that, in addition to our regular frequency quarterly PROFIT workfile page (p. 184), we have a regular frequency monthly page containing observations spanning the period from August 2002 to March 2003. Linking the PROFIT data from the quarterly page into the monthly page by date, with frequency conversion, requires that we specify an up-conversion method. Here, we show results of a frequency conversion link using both the simple constant-match average (PROFIT2) and quadratic-match average (PROFIT3) methods:

Month	Profit2	Profit3
Aug 2002	150	152.407
Sep 2002	150	144.630
Oct 2002	105	114.074
Nov 2002	105	103.519
Dec 2002	105	97.407
Jan 2003	100	97.222
Feb 2003	100	98.889
Mar 2003	100	103.889

Note that the PROFIT2 values are the same as those obtained by linking using simple date match merging, since the constant-match average method simply repeats the PROFIT observations for each matching month. Conversely, the PROFIT3 values are obtained using

an interpolation method that is only available for linking by date with frequency conversion.

Panel frequency conversion

There are additional issues to consider when performing frequency conversion links in panel workfile settings. When working with regular frequency panel pages, frequency conversion links construct values in the destination page in the following manner:

- If the source page is a regular frequency panel, we contract the source series by computing *means* across the panel identifiers. Note that means is the only contraction allowed. The result, which is a series that follows the source frequency, will be used as the source series.
- Next (*if necessary*), the source series is frequency converted to the destination page regular frequency using the series default conversion methods. If a conversion is performed, the frequency converted series becomes the new source series.
- We perform a one-to-one or one-to-many match merge of the source series into the destination page using exact date matching. A given source observation is *repeated* for all matching observations in the destination page. Repeated observations is the only match merge method allowed in this stage.

With frequency conversion linking, all date matching between pages is exact since we first contract the data to the source regular frequency and then perform a frequency conversion to the destination frequency. Only then do we perform a simple match merge of the data to the destination page.

An example will illustrate the general approach. Suppose again that we are working with the regular frequency, quarterly panel REVENUE data. For convenience, we repeat the data here:

Firm	Quarter	Revenue
1	2002Q1	120
1	2002Q2	130
1	2002Q3	150
1	2002Q4	105
1	2003Q1	100
1	2003Q2	125
1	2003Q3	200
1	2003Q4	170
2	2002Q1	40

2	2002Q2	40
2	2002Q3	50
2	2002Q4	35
2	2003Q1	20
2	2003Q2	25
2	2003Q3	50
2	2003Q4	40

We now wish to use frequency conversion to link these data into an annual panel by date, using the constant-match sum frequency conversion method.

The first step in resolving the frequency conversion link is to contract the source series to a regular quarterly frequency by taking averages across firms, yielding:

Quarter	Revenue
2002Q1	80
2002Q2	85
2002Q3	100
2002Q4	70
2003Q1	60
2003Q2	75
2003Q3	125
2003Q4	105

Next, the link frequency converts the quarterly series into an annual series using the specified frequency conversion methods. Since we have chosen to use the sum method, the frequency conversion aggregates the quarterly revenue, yielding:

Year	Revenue
2002	335
2003	365

Only after this frequency conversion step is completed do we perform the match merge of the annual data to the annual panel:

Firm	Year	Revenue2
1	2002	335
1	2003	365

2	2002	335
2	2003	365

Bear in mind that the first two steps, the averaging across firms to obtain a quarterly frequency series, and the frequency conversion to obtain an annual frequency series, are all performed automatically by the link, and are invisible to the user.

The results of frequency conversion linking from the quarterly panel to the annual panel differ significantly from the results obtained by general panel match merging using dates processing of matches. If we had performed the latter by creating a standard link by match merge with sum, we would have obtained:

Firm	Year	Revenue3
1	2002	670
1	2003	730
2	2002	670
2	2003	730

In creating a link that matches dates between the two panel workfile pages, we have a many-to-many match merge. In this case, the initial contraction involves summing over both quarters and firms to obtain annual values for 2002 (670) and 2003 (730). The second step, match merges these contracted values into the annual panel using a one-to-many match merge.

See "Panel links with date matching" on page 185 for related discussion.

Creating a Link

Links may be created interactively either by copying-and-pasting a series from the source to the destination page, or by issuing a link declaration in the destination page.

Creating a link using copy-and-paste

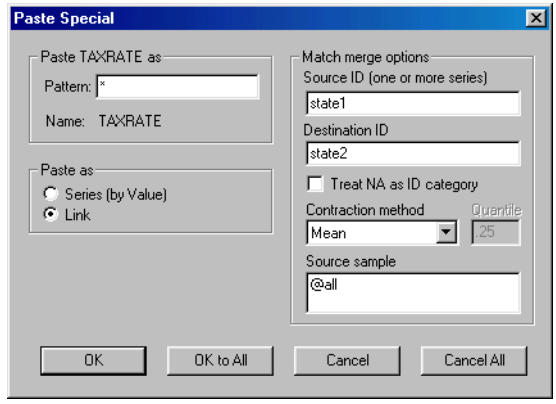
To define a link using copy-and-paste, first select one or more source series in the source workfile page, and either click on the right mouse button and select **Copy**, or select **Edit/Copy** from the main EViews menu. Next, switch to the destination page by clicking on the appropriate tab, and either click on the right mouse button and select **Paste Special...**, or select **Edit/Paste Special...** from the main menu.

General match merge links

If neither the source nor the destination series are dated pages, EViews will display a dialog prompting you to fill out the general match merge options. Here we have used **Paste**

Special... to copy-and-paste the series TAXRATE from the source page into a destination page.

Destination name



The field in the upper left-hand portion of the dialog should be used for specifying the name of the destination object. Here, we have the default wildcard value of "*" indicating that the series named TAXRATE in the source page will be used in the destination page. We may modify the name by typing an explicit name such as "NEWTAX", or by entering an expression containing the wildcard character. For example, if we wish to use the name "NEWTAXRATE" in the destination page, we may enter "NEW*" in the edit field.

The wildcard processing is particularly useful if you are copying multiple series into a new page since it facilitates batch renaming of series.

Destination type

Next, you will choose between pasting the series by value, or pasting the series as a link. If you paste by value, EViews will create an ordinary series in the destination page, and will fill it with the values from the link evaluation. If you paste your series as a link, EViews will create an actual link object containing the desired specification. As you might expect, there are significant differences between the two methods of copying your series.

In the first method, the link computations are performed immediately and the destination series values are assigned at the time the series is created. This behavior follows the traditional model of match merging and frequency conversion in which the operation is performed once to compute static values.

When you paste your series as a link, EViews defines a link object containing a specification of the match merge or frequency conversion. At creation, the link object is not evaluated and uses no memory. Then, whenever you access the values in the link series, EViews will determine whether the object needs evaluation and if so, will allocate memory and perform the link calculations.

With links, you gain the benefits of efficient memory use and dynamic updating of the values in the destination, at the cost of some speed since the link calculations may be performed more than once. Along these lines, it is worth pointing out that links may be

converted into ordinary series at any time. Once a series is created, however, it may not be converted back into a link.

Match merge options

Whether you elect to create a new series with fixed values or to create a new link series, you must specify link options.

Match ID information

First, you must specify the information that EViews will use to identify matches between observations in the two pages.

In the **Source ID** and **Destination ID** edit fields, you will enter the names of one or more source ID series and one or more destination ID series. The number and order of the names in the two fields should match. Thus, if you wish to match both CXID1 and PERIOD1 in the source page to CXID2 and PERIOD2 in the second page, you should enter the sets of names in parallel. Here, we choose to match observations using the values of the STATE1 series in the source page and the values of the STATE2 series in the destination page.

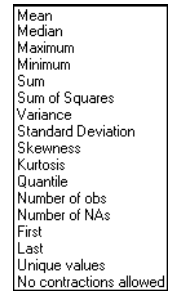
Next, there is a checkbox labeled **Treat NA as ID category** for whether to use observations which have NA values in the source and destination ID values. By default, observations are ignored if there are NAs in the ID series; by selecting this option, you instruct EViews to match observations with NA ID values from the source page to observations with NA ID values in the destination page.

Link calculation settings

The remaining options are used when computing the link values.

First, you should specify a source series contraction method. As described in "Linking by general match merging" on page 178, the first step in every match merge is to perform a contraction to ensure uniqueness of the source values. Since contraction is always performed, you should pay attention to your contraction method even when the source IDs are unique, since some settings will not yield the original source data.

There is an extensive list of contractions from which you may choose. For links involving numeric series you may choose to employ obvious methods such as the **Mean** (default) or the **Median** of the observations, or less obvious summary statistics such as the **Variance**, **Kurtosis**, **Quantile**, **Number of obs**, or **Number of NAs**.



For links involving alpha series, you must select from a subset of the numeric contractions: **Unique values** (default) , **No contractions allowed**, **First**, **Last**, **Maximum**, **Minimum**, **Number of obs**, **Number of NAs**.

Most of these options are self-explanatory, though a few comments about the choice of method may prove useful.

First, there are two options at the bottom of the list which deserve additional explanation. The last choice, **No contractions allowed,** may be used to ensure that contractions are never performed prior in the first step of a link match merge. The option is designed for cases where you believe that your source ID values are unique, and wish the link to generate an error if they are not.

The **Unique values** option provides a less strict version of the **No contractions allowed** setting, allowing for non-unique source ID values so long as any observations with matching IDs share the same source series value. In this case, the contraction will simply identify the unique source value associated with each unique source ID value. If there are observations with a single ID that have more than one source series value, the link will generate an error.

To see the difference between the two settings, note that contracting the following SOURCE and ID series

ID	Source
1	80
1	80
1	80
2	100
2	100

generates an error with the **Unique values** setting, but not with the **No contractions allowed** setting. Alternatively, the SOURCE and ID series

ID	Source
1	80
1	80
1	50
2	100
2	100

generate errors with both contractions.

Second, you should note that if you select **First** or **Last**, EViews will contract the source series by selecting the first or last observation in each set of observations with repeated source IDs. **First** or **Last** is defined here as depending on the order in which the observa-

tions appear in the original source workfile. Thus, selecting **First** means that the contracted value for each source ID value will be taken from the first observation in the workfile with that ID value.

Lastly, you should bear in mind that unless you select **No contractions allowed** or **Unique values**, EViews will perform a first stage contraction of the data using the specified settings. In cases where the source ID values are not unique, this contraction is a necessary step; in cases where the source ID values are unique, the contraction is not necessary for the resulting one-to-one or one-to-many match merge, but is performed so that EViews can support more complicated many-to-many merge operations.

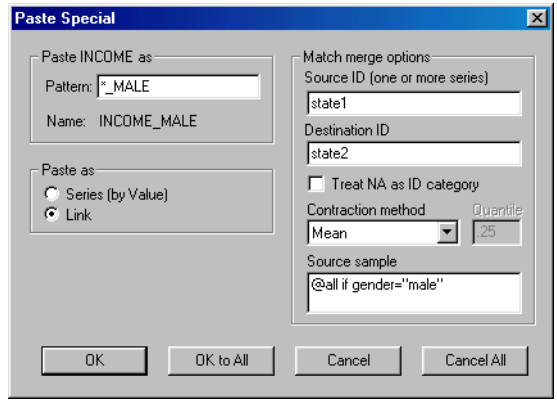
For most of the choices, performing a contraction on the unique source data has no practical effect on the outcome of a one-to-one or one-to-many match merge. For example, a choice of any of the data preserving options: **Mean**, **Median**, **Maximum**, **Minimum**, **Sum**, **First**, **Last**, **Unique values**, or **No contractions allowed** will create a link that performs the standard one-to-one or one-to-many match merge of the values of the original source series into the destination page.

On the other hand, selecting a contraction method that alters the source values will create a link that performs a match merge of the summary values into the destination page. Thus, selecting **Sum of Squares**, **Variance**, **Standard Deviation**, **Skewness**, **Kurtosis**, **Quantile**, **Number of obs**, or **Number of NAs**, will generate link values that differ from those obtained in a traditional one-to-one or one-to-many match merge.

It is worth emphasizing that the default contraction setting, **Mean,** preserves values for data with unique source IDs. Thus, unless you specifically set the contraction method to a non-preserving method, a one-to-one or one-to-many match merge will link the original values into the destination page. You may also ensure that EViews performs the traditional one-to-one or one-to-many match merge by selecting any of the other value preserving transformation methods, or even better by selecting **No contractions allowed** or **Unique values** to validate the IDs.

Finally, in the **Source sample** edit field, you should enter a description of the source sample to be used when constructing link values. By default, the full sample keyword "@ALL" is entered in the field so that EViews will use all of the observations in the source page.

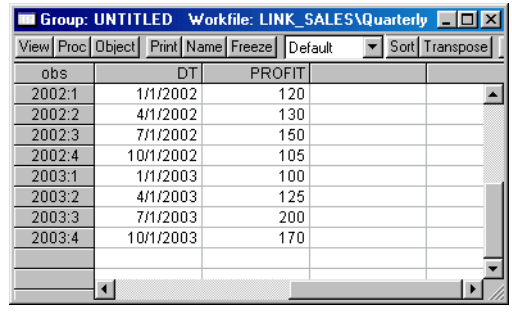
One important application involving sample settings is to restrict the observations over which the contraction is performed prior to performing the match merge. Suppose, for example, that we have a workfile with observations on individuals with state of residence. Then we could construct two links from the individual page to a state page, one of which computes the mean INCOME for males in each state, and another which computes the mean INCOME for females.



Date match merge links

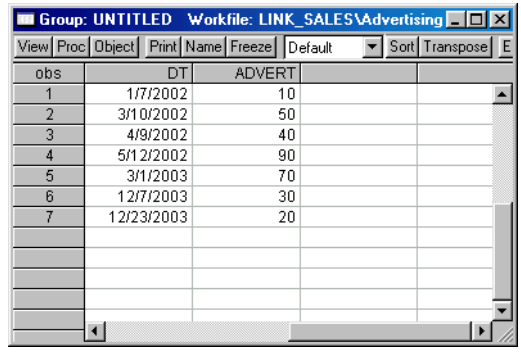
Dates may be used in matching in two ways: exact matching or date matching (see "Linking by date match merging" on page 183 for details).

Suppose we have a workfile containing the quarterly data on PROFITS described earlier. The quarterly PROFITS data is contained in a regular frequency quarterly workfile page. Also contained in the page is a date series DT generated by taking the first instance in each quarter ("series dt = @date"). We show here DT formatted to show the day-month-year, alongside the PROFIT series.



obs	DT	PROFIT
2002:1	1/1/2002	120
2002:2	4/1/2002	130
2002:3	7/1/2002	150
2002:4	10/1/2002	105
2003:1	1/1/2003	100
2003:2	4/1/2003	125
2003:3	7/1/2003	200
2003:4	10/1/2003	170

Contained in a separate, unstructured page are advertising data ADVERT, and another series DT showing the corresponding irregular dates.



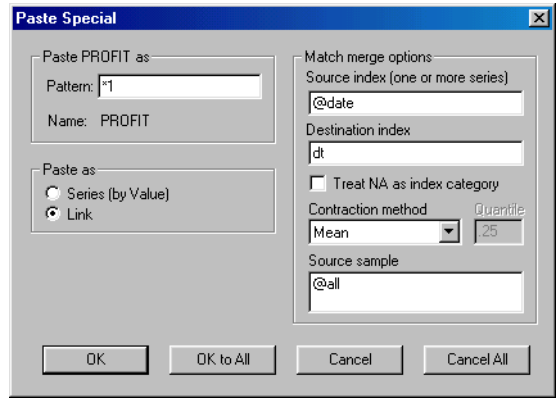
obs	DT	ADVERT
1	1/7/2002	10
2	3/10/2002	50
3	4/9/2002	40
4	5/12/2002	90
5	3/1/2003	70
6	12/7/2003	30
7	12/23/2003	20

If we attempt to match merge these data using the DT date series as identifiers, EViews will use the first method, exact matching, to identify common observations. Thus, if we try to link the PROFIT data into the advertising page using the

DT series as the identifiers, we will find that there are no observations in the quarterly source page that match observations in the irregular daily destination page. The resulting link values will all be NAs.

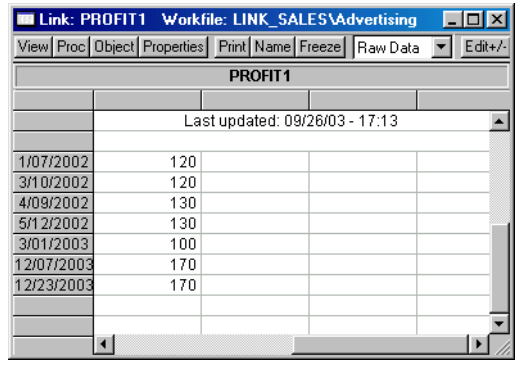
When one or both of the pages follow a regular frequency, we may instruct EViews to employ date matching. We may do so by using the special ID keyword "@DATE" as an ID in the regular frequency page ID to indicate that we wish to use date matching with the built-in date identifiers given by the structure of the page. In this case, we will use "@DATE" as the ID for the regular frequency quarterly page, and match it against the values in the DT series in the destination page.

In this example, we use the **Paste Special** dialog to instruct EViews to copy the quarterly PROFIT series to a link named PROFIT1 in the destination page. We employ date matching to match the quarters in the source page to the values in the DT series in the destination page, rounding to the lowest common frequency.



We first compute a **Mean** contraction of the source data for all observations, then match merge the contracted results into the destination. Note that since the match merge in this example is one-to-many, the **Mean** contraction method is irrelevant since it leaves the source data unchanged. If we wish to guarantee that the source IDs are unique, we may change the **Contraction** method to **No contractions allowed**.

In the special case where you have two dated structured pages, you may construct the link using the "@DATE" keyword for both page identifiers. Here, where the advertising page is structured as an (irregular) daily dated page, we could replace DT in the destination index field with the keyword "@DATE".



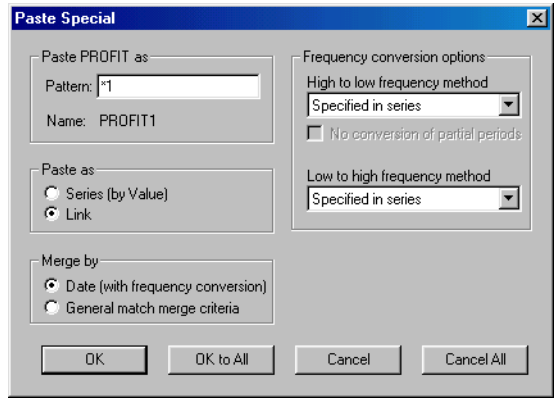
If "@DATE" is used as an ID in both pages, EViews will use the observation

date identifiers associated with the structure of each page, round them to the lowest common frequency, and then find matching observations.

Frequency conversion links

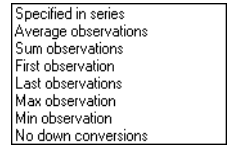
In the special case where we link numeric series between two regular frequency pages, we may copy-and-paste to define a link (or a by value copy of the source series) that employs frequency conversion ("Linking by date with frequency conversion" on page 187). In this setting, the **Paste Special** dialog offers you an additional choice between linking by general match merge, or linking by date using frequency conversion.

If you select **General match merge criteria** in the **Merge by** section of the dialog, the right side of the dialog with change to show the standard match merge version described in "General match merge links" on page 191.



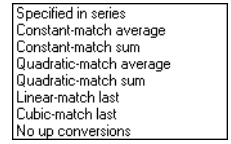
Alternately, to define a frequency conversion link, click on the **Date (with frequency conversion)** selection. The dialog will change to display the frequency conversion options for converting data both from high to low, and low to high frequency.

By default, EViews will use the high to low and the low to high conversion methods specified in the original source series.



If you wish to change the high to low conversion methods, simply select the desired setting from the drop-down menu. In addition, if you select one of the non-default methods, choose whether to select the **No conversion of partial periods** checkbox. If this setting is selected, EViews will propagate NAs when performing the frequency conversion so that the average of observations with an NA value will not drop the observation, and will instead generate an NA.

Note that the last conversion method, **No down conversions**, may be used to disallow down frequency conversion of the data. This setting allows you to ensure that when evaluated, the link involves same frequency (one-to-one) or low to high (one-to-many) frequency conversion, otherwise the link evaluation will generate an error.



To set the low to high conversion method, select the desired method from the drop-down menu. Once again, the last frequency conversion method, **No up conversions**, allows you to inform EViews that you expect the link to work only for same frequency, or high-to-low frequency linking, and that the link evaluation should generate an error if it encounters data requiring up conversion.

Creating a link by command

While the copy-and-paste interface is the easiest approach to specifying a link, we note that you may also create links using the LINK declaration statement and the LINKTO procedure.

You may, at the command line, enter the keyword "LINK" followed by the name of a new link object. EViews will create a new, incompletely specified, link object in the current (destination) workfile page. The destination page should be active when you enter the command.

You may modify a link specification, defining link IDs, as well as contraction and in some cases, expansion methods using the LINKTO proc.

Consider our earlier example where we link the TAXRATE data from the state page to the individual page. The following command creates a link object in the current workfile page:

```
link taxrate2
```

You may modify the TAXRATE2 link by providing a link definition using the LINKTO procedure. The "LINKTO" keyword should be followed by the name of the source series and the source and destination IDs, with the latter separated by "@SRC" and "@DEST" keywords. For example, if the link object TAXRATE2 exists in our individual page, the link proc:

```
taxrate2.linkto state::taxrate @src state1 @dest state2
```

instructs EViews to define the link TAXRATE2 so that it uses the TAXRATE series in the source page named "STATE" as the source series, and matches the source page STATE1 values to the current page STATE2 values.

In the special case where there is only one ID series in each page, we may, without introducing ambiguity, omit the "@SRC" and "@DEST" keywords. Here, we may shorten our link definition statement to:

```
taxrate2.linkto state::taxrate state1 state2
```

Lastly, we may combine these declaration and definition statements into one. The command

```
link taxrate2.linkto state::taxrate state1 state2
```

both creates a link object in the active workfile page and defines the source and link ID series.

In this one-to-many example where we link state data to individuals, we need not consider contraction methods as the default (mean) contraction method preserves the original data. If you wish to disallow contractions, or to limit them to cases where the values of the source data are unique, you may use contraction options as in:

```
link taxrate2.linkto(c=none) state::taxrate state1 state2
```

or

```
link taxrate2.linkto(c=unique) state::taxrate state1 state2
```

Conversely, linking the SALES data from the individual page to the state page yields a many-to-one conversion in which the contraction method is important. In this setting, we may optionally specify a contraction method so that when the state page is active, the statement

```
link sales2.linkto(c=sum) indiv::sales state2 state1
```

links the SALES data from the "INDIV" source page, matching the source page STATE2 values to the current page STATE1 values, and contracting observations using the sum transformation. If the contraction option is not provided, EViews will use the mean contraction default.

In the special case where you wish to link your data using date matching, you must use the special keyword "@DATE" as an ID series for the regular frequency page. For example, when linking from our quarterly to our advertising page, we may specify:

```
link profit1.linkto quarterly::profit @date dt
```

to tell EViews to link the quarterly page PROFIT data, matching the built-in identifier for the quarter with the date series DT in the destination advertising page.

As in the copy-and-paste interface, the presence of the special "@DATE" keyword tells EViews that you wish to perform date matching using the date structure of the corresponding regular frequency page. If "@DATE" is not specified as an ID, EViews will employ a general match merge using the specified identifiers.

When linking data between dated regular frequency workfile pages, the LINKTO proc will perform a frequency conversion link between the two pages *unless* ID series are explicitly provided, or a general match merge specific conversion method (such as variance or kurtosis) is specified. Thus, issuing the command

```
link profit2.linkto quarterly::profit
```

in an annual page, creates a frequency conversion link PROFIT2 using the PROFIT data from the quarterly page. Since no conversion options are provided, EViews will use the default frequency conversion method specified in the quarterly PROFIT series.

If ID series are provided, EViews will perform the link using general match merging. Thus, the closely related command

```
link profit2a.linkto quarterly::profit @date @date
```

will produce a link named PROFIT2A that employs date match merging using the dates in the workfile page structures. Since no conversion options are provided. EViews will use the default match merge contraction method, taking means, to perform the conversion.

If no ID series are specified, but a match merge specific option is provided, "@DATE @DATE" is appended to the ID list, and general match merging is assumed. Thus, the command

```
link profit2b.linkto(c=med) quarterly::profit
```

is equivalent to

```
link profit2b.linkto(c=med) quarterly::profit @date @date
```

since "c = med" is a match merge specific conversion option. This link is evaluated using general match merging, with date matching.

For additional details see `link` (p. 338) and `linkto` (p. 339) in the *Command and Programming Reference*.

Working with Links

Once a link is defined, you may, for all intents and purposes, use it as though it were an ordinary series or an alpha series.

Links may be identified in the workfile directory by the presence of a pink series or alpha series icon, or by an icon containing a "?". If a link definition uses an ordinary source series, it will appear in the workfile directory with a pink version of the series icon. If a link uses an alpha source series, it will appear with a pink alpha series icon. In both cases, the link may be used as though it were a series of the specified type.

If the link source series is not specified or if its type cannot be identified, the link icon will feature a "?" indicating that the link is unavailable. Undefined links will be classified as numeric series that generate NA values for every observation.

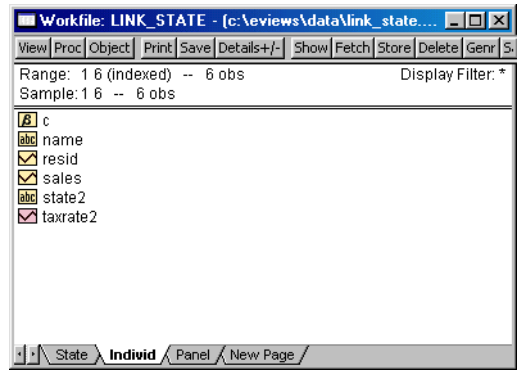
Using links

Links use virtually no memory until used. A link goes into use either when you are examining the contents of the link, when it is placed in a group which evaluates the link, or

when the link is used in a series expression. Once a link goes out of use, the memory for the link is cleared and made available for other uses. In this way, links take up only the minimum amount of memory required to perform a given operation.

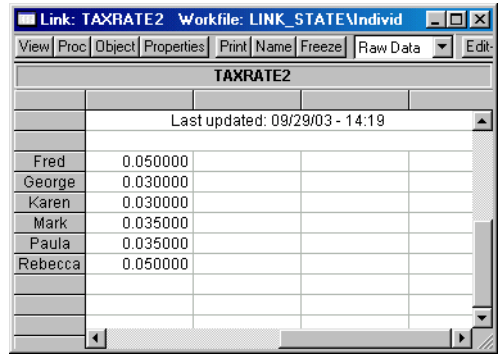
When links are in use, any modification to the data underlying the link will lead to a reevaluation of the link values. If you modify either the source series, or the source or destination ID series, EViews will automatically recompute the link values. In this way, you may use the link to define an automatically updating match merge or frequency conversion.

For example, suppose that we open a workfile containing the state and individual pages. Here we see the individual page containing the state TAXRATE data linked into the link series TAXRATE2. From the (colored) series icon, we see that TAXRATE2 is a link of a numeric series.



If the TAXRATE2 link is not in use, the link series contains no values and takes up no memory. Links are placed in use either by opening the link window, by placing the link in a group object, or by using the link in a series expression. Whenever the link comes into use, or one of the link components is changed, the link is evaluated, and its values updated as necessary.

For example, if we double click on the TAXRATE2 icon, we open a standard series spreadsheet view. At this point, EViews evaluates the link, performing the match merge operation, and assigning the values to the TAXRATE2 link. Note that the "Last updated" line will show the time that the link values were evaluated.



All of the menus and toolbars are those found in ordinary series—you may work with this link as though it were any ordinary series. Indeed, the only hint you will have that TAXRATE2 is not an ordinary series or alpha series is in the titlebar, which will indicate that we are working with a link object. For example, if you select **View/One-way tab-**

ulation... uncheck the grouping settings, and click on **OK** to continue, EViews will display a frequency tabulation of the contents of the link, just as it would for an ordinary series.

Tabulation of TAXRATE
Date: 09/29/03 Time: 15:27
Sample: 1 4
Included observations: 4
Number of categories: 4

Value	Count	Percent	Cumulative Count	Cumulative Percent
0.012000	1	25.00	1	25.00
0.030000	1	25.00	2	50.00
0.035000	1	25.00	3	75.00
0.050000	1	25.00	4	100.00
Total	4	100.00	4	100.00

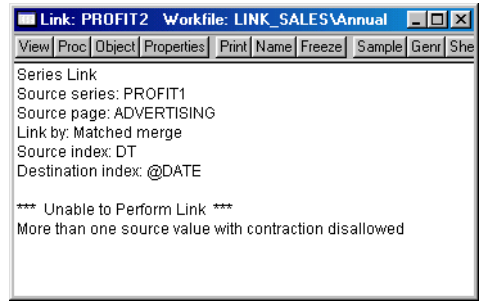
If you then close the link window, EViews will examine any open windows or existing group objects to see whether the link is still in use. If the link is no longer used, its contents will be cleared and memory will be released. The next time you use the link, it will come into use and will be reevaluated.

Similarly, you may use TAXRATE2 in any place that a series may be used. For example, we may generate a new series, TAXPCT that contains the values of TAXRATE2 expressed in percentage terms:

```
series taxpct = taxrate2 * 100
```

Assuming that TAXRATE2 is not currently in use, EViews will evaluate the link and assign values to each observation, then will multiply each of these values by 100 and assign them to TAXPCT. When the series assignment operation is completed, the values of TAXRATE2 will no longer be used, so that EViews will clear the link contents.

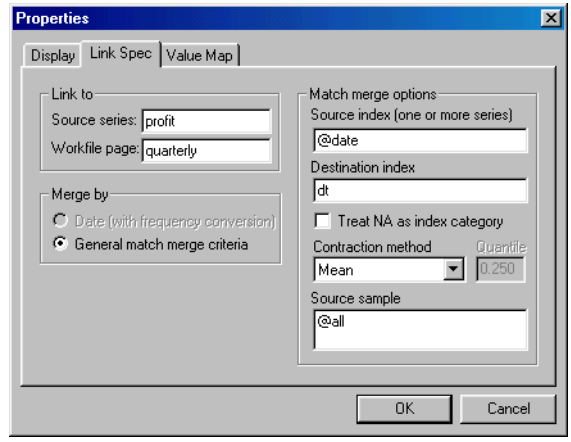
If you attempt to open a link that is improperly defined, either because the source or ID series are not found, or because the observed data require a contraction or frequency conversion method that has been disallowed, EViews will display a link view showing the definition of the link and the error encountered. If you attempt to use this link, you will find that all of the link values are set to NA.


Link: PROFIT2 Workfile: LINK_SALES\Annual
View Proc Object Properties Print Name Freeze Sample Genr She
Series Link
Source series: PROFIT1
Source page: ADVERTISING
Link by: Matched merge
Source index: DT
Destination index: @DATE
*** Unable to Perform Link ***
More than one source value with contraction disallowed

Modifying links

You may, at any time, modify the definition of a link by dialog or command.

To modify a link interactively, we must open the **Link Spec** dialog page. First open the desired link by double clicking on the icon in the workfile directory. Then click on the **Properties** toolbar button, or select **View/Properties...** from the main menu to bring up the link properties dialog. Lastly, select the **Link Spec** tab.



The **Link Spec** property page is a slightly modified version of the original **Paste Special** dialog used to create links. While the majority of the dialog is unchanged, in place of the destination name, we now have edit fields in which you specify the names of the source series and the source workfile page.

Here we see that the current link uses the PROFIT series in the QUARTERLY page as the source. The link is performed by general match merge, using date matching to link the quarterly dates to the destination series DT. The match merge first performs a mean contraction of the PROFIT series over the entire sample, and then performs the match merge.

To modify the link using the dialog, simply alter any of the dialog settings. For example, we may change the link contraction method from **Mean** to **Minimum** by changing the selection in the **Contraction method** combo box, or we may change the source sample by entering a new sample in the edit box. More fundamental changes in the link will result from changing the source series or workfile page, or any of the match merge identifiers.

To modify a link by command, you may use the LINKTO proc. See "Creating a link by command" on page 199 for details. Issuing a LINKTO proc command for an existing link will replace the existing values with the new specification.

Breaking links

The auto-updating feature is one of the most important characteristics of links. Given the live nature of links, changes to either the source series, or the source or destination IDs will lead EViews to recalculate the values of the link. Links may be used to create auto-updating match merges or frequency conversion of series between workfile pages.

Suppose, for example, that while displaying the TAXRATE2 spreadsheet view, you elect to edit the values in the individual STATE2 ID series. Changing Mark's value for STATE2 from "Texas" to "Arkansas" changes the values of an ID series used to compute the values in TAXRATE2. EViews automatically recomputes TAXRATE2, changing the value for Mark from 0.35 to 0.30, and updates the open spreadsheet view accordingly. Furthermore, any future use of the TAXRATE2 link will use the updated values.

In some circumstances, you may wish to fix the values of the link so that future changes to the source or ID series do not alter the existing values. There are two ways in which you may achieve this result. First, you may simply generate a new series that contains the current values of the link, as in:

```
series fixrate = taxrate2
```

The new ordinary series FIXRATE contains the current values of TAXRATE2. Furthermore, FIXRATE remains unchanged in the face of changes in TAXRATE2. With this method, both the original link series and a new series will be kept in the workfile.

The second method for fixing values is to convert the link into a series. We term this process *unlinking* or *breaking the link*. In this case, the existing link is replaced by a series with the same name, containing the values in the link at the time the link is broken. To break a link, simply select **Object/Unlink...**. EViews will prompt you to continue. Click on **OK** to proceed, bearing in mind that the process of unlinking is irreversible.

Chapter 9. Advanced Workfiles

In Chapter 3, "Workfile Basics", on page 49, we described the basics of workfiles; how to create and work with a workfile, as well as the basics of using multi-page workfiles. In this chapter, we describe advanced workfile types and tools for working with workfiles.

First, we describe the fundamentals of structured workfiles. You will need to understand the concepts underlying structured workfiles to work with irregular dated data, data involving cross-section identifiers, or panel structures.

Next, we outline various workfile level tools for managing your data. Among other things, we discuss the basics of resizing a workfile, saving a workfile to foreign formats, subsetting a workfile, and rearranging the format of the data in your workfile.

Structuring a Workfile

You may, at any time, change the underlying structure of an existing workfile or workfile page by applying structuring information. We call this process *structuring a workfile*. There are four primary types of structuring information that you may provide:

- regular date descriptions.
- variables containing observation identifiers for dated data.
- variables containing observation identifiers for cross-section data.
- variables containing observation identifiers defining a panel data structure.

Applying structures to the data in your workfiles was not possible in versions of EViews prior to EViews 5. The ability to structure your data is an important innovation, and we will explore structured workfiles at some length.

Types of Structured Data

Before describing the process of structuring a workfile or workfile page, we define some concepts related to the various data structures.

Regular and Irregular Frequency Data

As the name suggests, *regular frequency data* arrive at regular intervals (daily, monthly, annually, etc.). Standard macroeconomic data such as quarterly GDP or monthly housing starts are examples of regular frequency data. This type of data is introduced in "Creating a Workfile by Describing its Structure" on page 51.

Unlike regular frequency data, *Irregular frequency data* do not arrive in a precisely regular pattern. An important example of irregular data is found in stock and bond prices, where the presence of missing days due to holidays and other market closures means that the data do not follow a regular daily (7- or 5-day) frequency.

The most important characteristic of regular data is that there are no *structural gaps* in the data—all observations in the specified frequency *exist*, even if there are missing values that are not observed. Alternatively, irregular data allow for gaps between successive observations in the given regular frequency. This is a subtle distinction, but has important consequences for lag processing.

The distinction is best illustrated by an example. Suppose that we are working with a daily calendar and that we have two kinds of data: data on bond prices (BOND), and data on temperature in Los Angeles in Farenheit (TEMP):

Day	Day of Week	Bond	Temp
12/21	Sun	< mkt.closed >	68
12/22	Mon	102.78	70
12/23	Tues	102.79	NA
12/24	Wed	102.78	69
12/25	Thurs	< mkt.closed >	68
12/26	Fri	102.77	70

Notice that in this example, the bond price is not available on 12/21 and 12/25 (since the market was closed), and that the temperature reading was not available on 12/23 (due to equipment malfunction).

Typically, we would view the TEMP series as following a 7-day *regular* daily frequency with a missing value for 12/23. The key feature of this interpretation is that the day 12/23 *exists*, even though a temperature reading was not taken on that day. Most importantly, this interpretation implies that the lagged value of TEMP on 12/24 (the previous day's TEMP value) is NA.

In contrast, most analysts would view BOND prices as following an *irregular* daily frequency in which days involving market closures *do not exist*. Under this interpretation, we would remove weekends and holidays from the calendar so that the bond data would be given by:

Day	Day of Week	Bond
12/22	Mon	102.78
12/23	Tue	102.79
12/24	Wed	102.78
12/26	Fri	102.77

The central point here is that lags are defined differently for regular and irregular data. Given a regular daily frequency, the lagged value of BOND on 12/26 would be taken from the previous day, 12/25, and would be NA. Given the irregular daily frequency, the lagged value on 12/26 is taken from the previous observation, 12/24, and would be 102.78. In defining an irregular calendar, we explicitly skip over the structural gaps created by market closure.

You may always convert irregular frequency data into regular frequency data by adding any observations required to fill out the relevant calendar. If, for example, you have 7-day irregular data, you may convert it to a regular frequency by adding observations with IDs that correspond to any missing days.

Undated Data with Identifiers

Perhaps the simplest data structure involves undated data. We typically refer to these data as *cross-section data*. Among the most common examples of cross-section data are state data taken at a single point in time:

Obs	Year	State	TaxRate
1	2002	Alabama	.000
2	2002	Arkansas	.035
3	2002	Arizona	.035
...	2002	...	...
50	2002	Wyoming	.010

Here we have an alphabetically ordered dataset with 50 observations on state tax rates. We emphasize the point that these data are undated since the common YEAR of observation does not aid in identifying the individual observations.

These cross-section data may be treated as an unstructured dataset using the default integer identifiers 1 to 50. Alternatively, we may structure the data using the unique values in STATE as identifiers. These state name IDs will then be used when referring to or labeling observations. The advantages of using the state names as identifiers should be obvious—

comparing data for observation labeled "Arizona" and "Wyoming" is much easier than comparing data for observations "3" and "50".

One last comment about the ordering of observations in cross-section data. While we can (and will) define the lag observation to be that "preceding" a given observation, such a definition is sensitive to the arbitrary ordering of our data, and may not be meaningful. If, as in our example, we order our states alphabetically, the first lag of "Arkansas" is taken from the "Arizona" observation, while if we order our observations by population, the lag of "Arkansas" will be the data for "Utah".

Panel Data

Some data involve observations that possess both cross-section (*group*) and within-cross-section (*cell*) identifiers. We will term these to be *panel data.* Many of the previously encountered data structures may be viewed as a trivial case of panel data involving a single cross-section.

To extend our earlier example, suppose that instead of observing the cross-section state tax data for a single year, we observe these rates for several years. We may then treat an observation on any single tax rate as having two identifiers: a single identifier for STATE (the group ID), and an identifier for the YEAR (the cell ID). The data for two of our states, "Kansas" and "Kentucky" might look like the following:

Obs	State	Year	TaxRate
…	…	…	…
80	Kansas	2001	.035
81	Kansas	2002	.037
82	Kansas	2003	.036
83	Kentucky	2001	.014
84	Kentucky	2003	.016
…	…	…	…

We emphasize again that identifiers must uniquely determine the observation. A corollary of this requirement is that the *cell IDs uniquely identify observations within a group.* Note that requiring cell IDs to be unique within a group does not imply that the cell IDs are unique. In fact, cell ID values are usually repeated across groups; for example, a given YEAR value appears in many states since the tax rates are generally observed in the same years.

If we observe repeated values in the cell identifiers within *any* one group, we must either use a different cell identifier, or we must redefine our notion of a group. Suppose, for example, that Kansas changed its tax rate several times during 2002:

Obs	State	Year	Cell_ID1	Cell_ID2	TaxRate
…	…	…	…	…	…
80	Kansas	2001	1	1	.035
81	Kansas	2002	2	1	.037
82	Kansas	2002	3	2	.038
83	Kansas	2002	4	3	.035
84	Kansas	2003	5	1	.036
85	Kentucky	2001	1	1	.014
86	Kentucky	2003	2	2	.016
…	…	…	…	…	…

In this setting, YEAR would not be a valid cell ID for groups defined by STATE, since 2002 would be repeated for STATE = "Kansas".

There are a couple of things we may do. First, we may simply choose a different cell identifier. We could, for example, create a variable containing a default integer identifier running within each cross-section. For example, the newly created variable CELL_ID1 is a valid cell ID since it provides each "Kansas" and "Kentucky" observation with a unique (integer) value.

Alternately, we may elect to subdivide our groups. We may, for example, choose to use *both* STATE *and* YEAR as the group identifier. This specification defines a group for each unique STATE and YEAR combination (*e.g.* — observations for which STATE = "Kansas" *and* YEAR = "2002" would comprise a single group). Given this new group definition, we may use *either* CELL_ID1 or CELL_ID2 as cell identifiers since they are both unique for each STATE and YEAR group. Notice that CELL_ID2 could not have been used as a valid cell ID for STATE groups since it does not uniquely identify observations within Kansas.

While it may at first appear to be innocuous, the choice between creating a new variable or redefining your groups has important implications (especially for lag processing). Roughly speaking, if you believe that observations within the original groups are closely "related", you should create a new cell ID; if you believe that the subdivision creates groups that are more alike, then you should redefine your group IDs.

In our example, if you believe that the observations for "Kansas" in "2001" and "2002" are both fundamentally "Kansas" observations, then you should specify a new cell ID. On the other hand, if you believe that observations for "Kansas" in "2002" are very different from "Kansas" in "2001", you should subdivide the original "Kansas" group by using both STATE and YEAR as the group ID. The implications of this choice are explored in greater depth in "Lags, Leads, and Panel Structured Data" on page 212.

Lags, Leads, and Panel Structured Data

Following convention, the observations in our panel dataset are always *stacked by cross-section*. We first collect the observations by cross-section and sort the cell IDs within each cross-section. We then stack the cross sections on top of one another, with the data for the first cross-section followed by the data for the second cross-section, the second followed by the third, and so on.

The primary impact of this data arrangement is its effect on lag processing. There are two fundamental principles of lag processing in panel data structures:

- First, lags and leads do not cross group boundaries, so that they never use data from a different group.
- Second, lags and leads taken within a cross-section are defined over the sorted values of the cell ID. This implies that lags of an observation are always associated with lower value of the cell ID, and leads always involve a higher value (the first lag observation has the next lowest value and the first lead has the next highest value).

Let us return to our original example with STATE as the group ID and YEAR as the cell ID, and consider the values of TAXRATE, TAXRATE(-1), and TAXRATE(1). Applying the two rules for panel lag processing, we have:

Obs	State	Year	TaxRate	TaxRate(-1)	TaxRate(1)
…	…	…	…	…	…
80	Kansas	2001	.035	NA	.037
81	Kansas	2002	.037	.035	.036
82	Kansas	2003	.036	.037	NA
83	Kentucky	2001	.014	NA	.016
84	Kentucky	2003	.016	.014	NA
…	…	…	…		

Note in particular, that the lags and leads of TAXRATE do not cross the group boundaries; the value of TAXRATE(-1) for Kentucky in 2001 is an NA since the previous value is from Kansas, and the value TAXRATE(1) for Kansas in 2003 is NA is the next value is from Kentucky.

Next, consider an example where we have invalid IDs since there are duplicate YEAR values for Kansas. Recall that there are two possible solutions to this problem: (1) creating a new cell ID, or (2) redefining our groups. Here, we see why the choice between using a new cell ID or subdividing groups has important implications for lag processing. First, we may simply create a new cell ID that enumerates the observations in each state (CELL_ID1). If we use CELL_ID1 as the cell identifier, we have:

Obs	State	Year	Cell_ID1	TaxRate	TaxRate(-1)
…	…	…	…	…	
80	Kansas	2001	1	.035	NA
81	Kansas	2002	2	.037	.035
82	Kansas	2002	3	.038	.037
83	Kansas	2002	4	.035	.038
84	Kansas	2003	5	.036	.035
85	Kentucky	2001	1	.014	NA
86	Kentucky	2003	2	.016	.014
…	…	…	…	…	

Note that the only observations for TAXRATE(-1) that are missing are those at the "seams" joining the cross-sections.

Suppose instead that we elect to subdivide our STATE groupings by using both STATE and YEAR to identify a cross-section, and we create CELL_ID2 which enumerates the observations in each cross-section. Thus, each group is representative of a unique STATE-YEAR pair, and the cell ID indexes observations in a given STATE for a specific YEAR. The TAXRATE(-1) values are given in:

Obs	State	Year	Cell_ID2	TaxRate	TaxRate(-1)
…	…	…	…	…	…
80	Kansas	2001	1	.035	NA
81	Kansas	2002	1	.037	NA
82	Kansas	2002	2	.038	.037
83	Kansas	2002	3	.035	.038
84	Kansas	2003	1	.036	NA
85	Kentucky	2001	1	.014	NA
86	Kentucky	2003	2	.016	.014
…	…	…	…	…	…

Once again, the missing observations for TAXRATE(-1) are those that span cross-section boundaries. Note however, that since the group boundaries are now defined by STATE and YEAR, there are more seams and TAXRATE(-1) has additional missing values.

In this simple example, we see the difference between the alternate approaches for handling duplicate IDs. Subdividing our groups creates additional groups, and additional seams between those groups over which lags and leads are not processed. Accordingly, if

you wish your lags and leads to span all of the observations in the original groupings, you should create a new cell ID to be used with the original group identifier.

Types of Panel Data

Panel data may be characterized in a variety of ways. For purposes of creating panel workfiles in EViews, there are several concepts that are of particular interest.

Dated vs. Undated Panels

We characterize panel data as *dated* or *undated* on the basis of the cell ID. When the cell ID follows a frequency, we have a *dated panel* of the given frequency. If, for example, our cell IDs are defined by a variable like YEAR, we say we have an *annual panel*. Similarly, if the cell IDs are quarterly or daily identifiers, we say we have a quarterly or daily panel.

Alternatively, an *undated panel* uses group specific default integers as cell IDs; by default the cell IDs in each group are usually given by the default integers (1, 2, ...).

Regular vs. Irregular Dated Panels

Dated panels follow a regular or an irregular frequency. A panel is said to be a *regular frequency panel* if the cell IDs for every group follow a regular frequency. If one or more groups have cell ID values which do not follow a regular frequency, the panel is said to be an *irregular frequency panel*.

One can convert an irregular frequency panel into a regular frequency panel by adding observations to remove gaps in the calendar for all cross-sections. Note that this procedure is a form of internal balancing (see "Balanced vs. Unbalanced Panels" below) which uses the calendar to determine which observations to add instead of using the set of cell IDs found in the data.

See "Regular and Irregular Frequency Data" on page 207 for a general discussion of these topics.

Balanced vs. Unbalanced Panels

If every group in a panel has an identical set of cell ID values, we say that the panel is *fully balanced*. All other panel datasets are said to be *unbalanced*.

In the simplest form of balanced panel data, every cross-section follows the same regular frequency, with the same start and end dates—for example, data with 10 cross-sections, each with annual data from 1960 to 2002. Slightly more complex is the case where every cross-section has an identical set of irregular cell IDs. In this case, we say that the panel is balanced, but irregular.

We may balance a panel by adding observations to the unbalanced data. The procedure is quite simple—for each cross-section or group, we add observations corresponding to cell IDs that are not in the current group, but appear elsewhere in the data. By adding observa-

tions with these "missing" cell IDs, we ensure that all of the cross-sections have the same set of cell IDs.

To complicate matters, we may *partially* balance a panel. There are three possible methods—we may choose to balance between the starts and ends, to balance the starts, or to balance the ends. In each of these methods, we perform the procedure for balancing data described above, but with the set of relevant cell IDs obtained from a subset of the data. Performing all three forms of partial balancing is the same as fully balancing the panel.

Balancing data *between the starts and ends* involves adding observations with cell IDs that are not in the given group, but are both observed elsewhere in the data and lie between the start and end cell ID of the given group. If, for example, the earliest cell ID for a given group is "1985m01" and the latest ID is "1990m01", the set of cell IDs to consider adding is taken from the list of observed cell IDs that lie between these two dates. The effect of balancing data between starts and ends is to create a panel that is *internally balanced,* that is, balanced for observations with cell IDs ranging from the *latest* start cell ID to the *earliest* end cell ID.

A simple example will better illustrate this concept. Suppose we begin with a two-group panel dataset with the following data for the group ID (INDIV), and the cell ID (YEAR):

Indiv	Year	Indiv	Year
1	1985	2	1987
1	1987	2	1989
1	1993	2	1992
1	1994	2	1994
1	1995	2	1997
1	1996	2	2001

For convenience, we show the two groups side-by-side, instead of stacked. As depicted, these data represent an unbalanced, irregular, annual frequency panel. The data are unbalanced since the set of observed YEAR identifiers are not common for the two individuals; *i.e.* — "1985" appears for individual 1 (INDIV = "1"), but does not appear for individual 2 (INDIV = "2"). The data are also irregular since there are gaps in the yearly data for both individuals.

To balance the data between starts and ends, we first consider the observations for individual 1. The earliest cell ID for this cross-section is "1985" and the latest is "1996". Next, we examine the remainder of the dataset to obtain the cell IDs that lie between these two values. This set of IDs is given by {"1987", "1989", "1992", "1994"}. Since "1989" and "1992" do not appear for individual 1, we add observations with these two IDs to that cross-section. Likewise, for group 2, we obtain the cell IDs from the remaining data that lie

between “1987” and “2001”. This set is given by {“1993”, “1994”, “1995”, “1996”}. Since “1993”, “1995”, and “1996” do not appear for individual 2, observations with these three cell IDs will be added for individual 2.

The result of this internal balancing is an expanded, internally balanced panel dataset containing:

Indiv	Year	Indiv	Year
1	1985	2	—
1	1987	2	1987
1	*1989	2	1989
1	*1992	2	1992
1	1993	2	*1993
1	1994	2	1994
1	1995	2	*1995
1	1996	2	*1996
1	—	2	1997
1	—	2	2001

We have marked the five added observations with an asterisk, and arranged the data so that the cell IDs line up where possible. Observations that are not present in the dataset are marked as “—”. Notice that the effect of the internal balancing is to fill in the missing cell IDs in the central portion of the data.

It is worth a digression to note here that an alternative form of internal balancing is to add observations to remove all gaps in the calendar between the starts and ends. This method of balancing, which converts the data from an irregular to a regular panel, uses the calendar to determine which observations to add instead of using the set of observed cell IDs found. If we are balancing the expanded dataset, we would add observations with the cell IDs for missing years: {“1986”, “1988”, “1990”, “1991”} for individual 1, and {“1988”, “1990”, “1991”, “1998”, “1999”, “2000”} for individual 2.

Lastly, we consider the effects of choosing to *balance the starts* or *balance the ends* of our data. In the former case, we ensure that every cross-section adds observations corresponding to observed cell IDs that come before the current starting cell ID. In this case, balancing the starts means adding an observation with ID “1985” to group 2. Similarly, balancing the ends ensures that we add, to every cross-section, observations corresponding to observed cell IDs that follow the cross-section end cell ID. In this case, balancing the ends involves adding observations with cell IDs “1997” and “2001” to group 1.

Nested Panels

While cell IDs must uniquely identify observations within a group, they typically contain values that are repeated across groups. A *nested panel* data structure is one in which the cell IDs are nested, so that they are unique both within and across groups. When cell IDs are nested, they uniquely identify the individual observations in the dataset.

Consider, for example, the following nested panel data containing identifiers for both make and model of automobile:

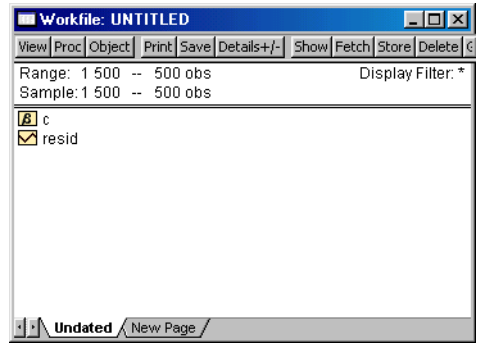
Make	Model
Chevy	Blazer
Chevy	Corvette
Chevy	Astro
Ford	Explorer
Ford	Focus
Ford	Taurus
Ford	Mustang
Chrysler	Crossfire
Chrysler	PT Cruiser
Chrysler	Voyager

We may select MAKE as our group ID, and MODEL as our cell ID. MODEL is a valid cell ID since it clearly satisfies the requirement that it uniquely identify the observations within each group. MODEL is also nested within MAKE since each cell ID value appears in exactly one group. Since there are no duplicate values of MODEL, it may be used to identify every observation in the dataset.

There are a number of complications associated with working with nested panel data. At present, EViews does not allow you to define a nested panel data structure.

Applying a Structure to a Workfile

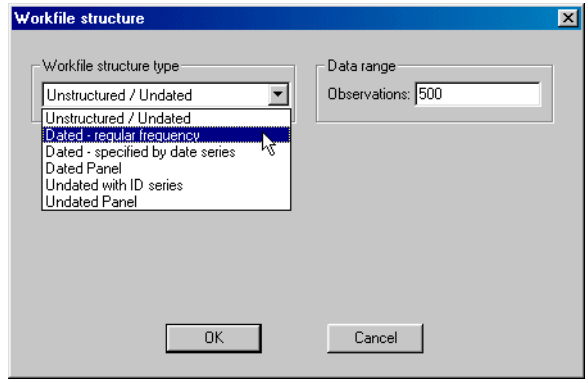
To structure an existing workfile, select **Proc/"Structure/Resize Current Page..."** in the main workfile window, or double-click on the portion of the window displaying the current range ("Range:").



Selecting a Workfile Type

EViews opens the **Workfile structure** dialog. The basic structure of the dialog is quite similar to the **Workfile create** dialog ("Creating a Workfile" on page 50). On the left-hand side is a combo box where you will select a structure type.

Clicking on the structure type combo box brings up several choices. As before, you may choose between the **Unstructured/Undated**, and **Dated - regular frequency** types. There are, however, several new options. In the place of **Balanced Panel**, you have the option to select from **Dated - specified by date series**, **Dated Panel**, **Undated with ID series**, or **Undated Panel**.

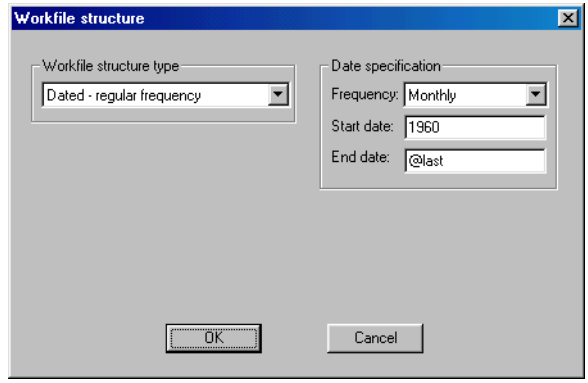


Workfile Structure Settings

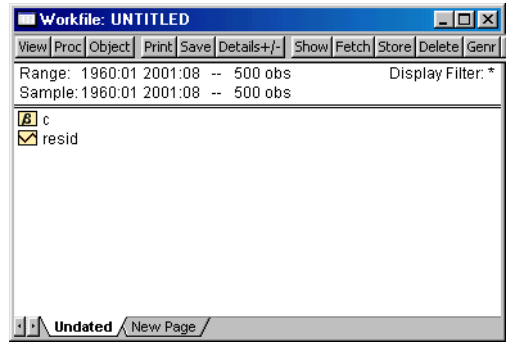
As you select different workfile structure types, the right-hand side of the dialog changes to show relevant settings and options for the selected type. For example, if you select the **Dated - regular frequency type**, you will be prompted to enter information about the frequency of your data and date information; if you select an **Undated Panel**, you will be prompted for information about identifiers and the handling of balancing operations.

Dated - Regular Frequency

Given an existing workfile, the simplest method for defining a regular frequency structured workfile is to select **Dated - regular frequency** in the structure type combo box. The right side of the dialog changes to reflect your choice, prompting you to describe your data structure.



You are given the choice of a **Frequency**, as well as a **Start date** and **End date**. The only difference between this dialog and the workfile create version is that the **End date** field is prefilled with "@LAST". This default reflects the fact that given a start date and the number of observations in the existing workfile, EViews can calculate the end date implied by "@LAST". Alternatively, if we provide an ending date, and enter "@FIRST" in the **Start date** field, EViews will automatically calculate the date associated with "@FIRST".

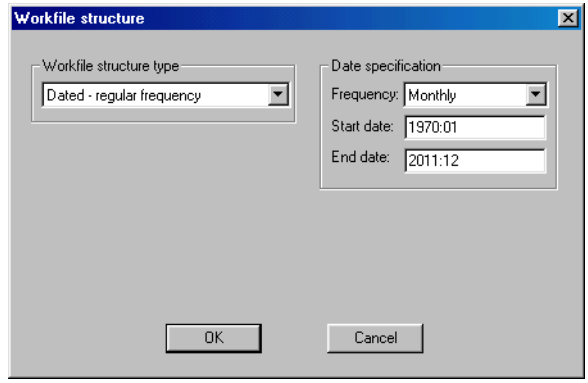


If we fill out the desired fields and click on **OK**, EViews will restructure the workfile. In this example, we have specified a monthly frequency starting in 1960:01 and continuing until "@LAST". There are exactly 500 observations in the workfile since the end date was calculated to match the existing workfile size.

Alternatively, we might elect to enter explicit values for both the starting and ending dates. In this case, EViews will calculate the number of observations implied by these dates and the specified frequency. If the number does not match the number of observations in the existing workfile, you will be informed of this fact, and prompted to continue. If you choose to proceed, EViews will both restructure and resize the workfile to match your specification.

One consequence of this behavior is that resizing a workfile is a particular form of restructuring. To resize a workfile, simply call up the **Workfile structure** dialog, and change the beginning or ending date.

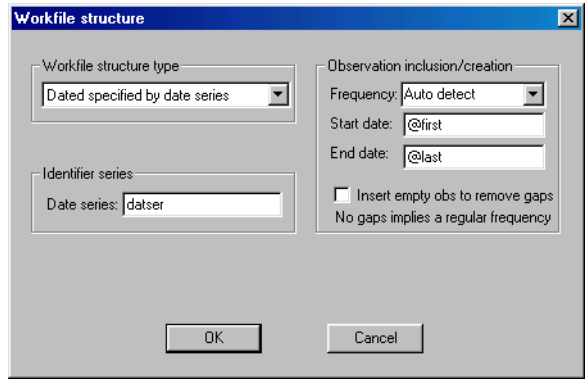
Here we have changed the **End date** from "2011:08" to "2011:12", thereby instructing EViews to add 4 observations to the end of the workfile. If you select **OK**, EViews will inform you that it will add 4 observations and prompt you to continue. If you proceed, EViews will resize the workfile to your specification.



Dated - specified by date series

The second approach to structuring your workfile is to provide the name of a series containing the dates (or series than may be interpreted as dates) to be used as observation identifiers. Select **Dated - specified by date series** in the combo box, and fill out the remainder of the dialog.

The first thing you must do is enter the name of one or more **Date series** that describe the unique date identifiers.



The series may contain EViews date values (a true date series), or the single or multiple series may contain numeric or string representations of unique dates. In the latter case, EViews will create a single date series containing the date values associated with the numeric or string representations. This new series, which will be given a name of the form DATEID##, will be used as the identifier series.

On the right side of the dialog, you will specify additional information about your workfile structure. In the first combo box, you will choose one of the standard EViews workfile frequencies (annual, quarterly, monthly, etc.). As shown in the image, there is an additional (default) option, **Auto detect**, where EViews attempts to detect the frequency of your data from the values in the specified series. In most cases you should use the default; if, however, you choose to override the auto-detection, EViews will associate the date values in the series with observations in the specified frequency.

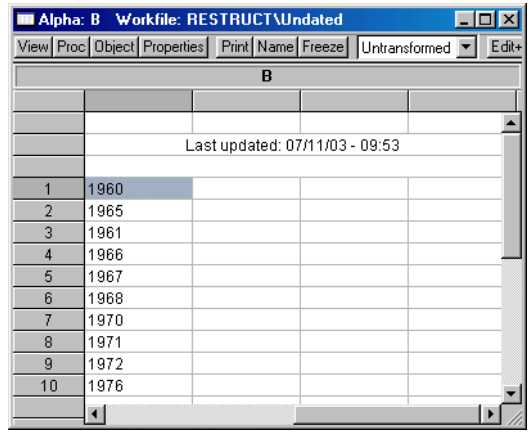
You may elect to use the EViews defaults, "@FIRST" and "@LAST", for the **Start date** and the **End date**. In this case, the earliest and latest dates found in the identifier series will be used to define the observations in the workfile. Alternatively, you may specify the start and end dates explicitly. If these dates involve resizing the workfile, you will be informed of this fact, and prompted to continue.

The last option is the **Insert empty obs** checkbox. This option should be used if you wish to ensure that you have a regular frequency workfile. If this option is selected, EViews will add any observations necessary to remove gaps in the calendar at the given frequency. If the option is not selected, EViews will use only the observed IDs in the workfile and the workfile may be structured as an irregular workfile.

Suppose, for example, that you have observation with IDs for the quarters 1990Q1, 1990Q2, 1990Q4, but not 1990Q3. If **Insert empty obs** is checked, EViews will remove the gap in the calendar by adding an observation corresponding to 1990:3. The resulting workfile will be structured as a regular quarterly frequency workfile. If you do not insert observations, the workfile will be treated as an irregular quarterly workfile.

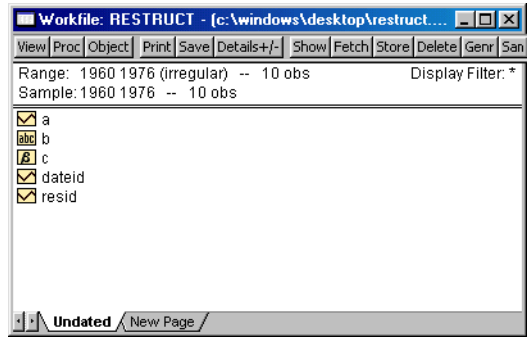
Once you click on **OK**, EViews will first look for duplicate observation IDs. If duplicates are not found, EViews will sort the data in your workfile by the values in the date series and define the specified workfile structure. In addition, the date series is locked so that it may not be altered, renamed, or deleted so long as it is being used to structure the workfile.

To illustrate the process of structuring a workfile by an ID series, we consider a simple example involving a 10 observation unstructured workfile.



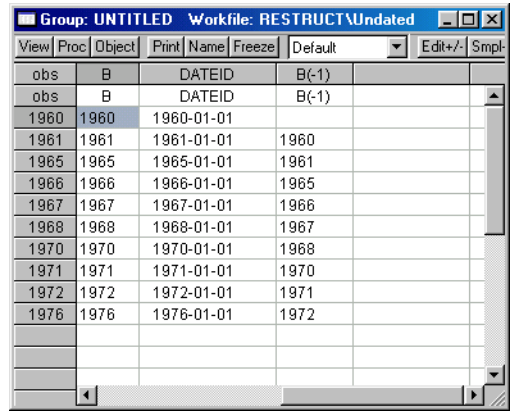
Suppose that the workfile contains the alpha series B consisting of string representations of dates, as depicted. The first thing you should notice about B is that the years are neither complete, nor ordered—there is, for example, no "1962", and "1965" precedes "1961". You should also note that since we have an unstructured workfile, the observation identifiers used to identify the rows of the table are given by the default integer values.

From the workfile window we call up the **Workfile structure** dialog, select **Dated - specified by date series** as our workfile type, and enter the name "B" in the **Date series** edit box. We will start by leaving all of the other settings at their defaults: the frequency is set at **Auto detect**, and the start and end dates are given by "@FIRST" and "@LAST".



The resulting (structured) workfile window shown here indicates that we have a 10 observation irregular annual frequency workfile that ranges from an earliest date of 1960 to the latest date of 19706

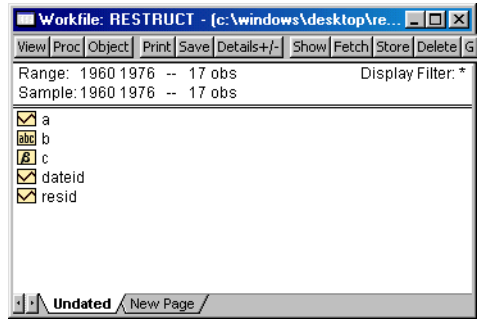
Since the series B contained only text representations of dates, EViews has created a new series DATEID containing date values corresponding to those in B. DATEID is locked and cannot be altered, renamed, or deleted so long as it is used to structure the workfile.



obs	B	DATEID	B(-1)
obs	B	DATEID	B(-1)
1960	1960	1960-01-01	
1961	1961	1961-01-01	1960
1965	1965	1965-01-01	1961
1966	1966	1966-01-01	1965
1967	1967	1967-01-01	1966
1968	1968	1968-01-01	1967
1970	1970	1970-01-01	1968
1971	1971	1971-01-01	1970
1972	1972	1972-01-01	1971
1976	1976	1976-01-01	1972

Here, we show a group containing the original series B, the new series DATEID, and the lag of B, B(-1). There are a few things to note. First, the observation identifiers are no longer integers, but instead are values taken from the identifier series DATEID. The formatting of the observation labels will use the display formatting present in the ID series. If you wish to change the appearance of the labels, you should set the display format for DATEID (see "Display Formats" on page 89).

Second, since we have sorted the contents of the workfile by the ID series, the values in B and DATEID are ordered by date. Third, the lagged values of series use the irregular calendar defined by DATEID—for example, the lag of the 1965 value is given by 1961.

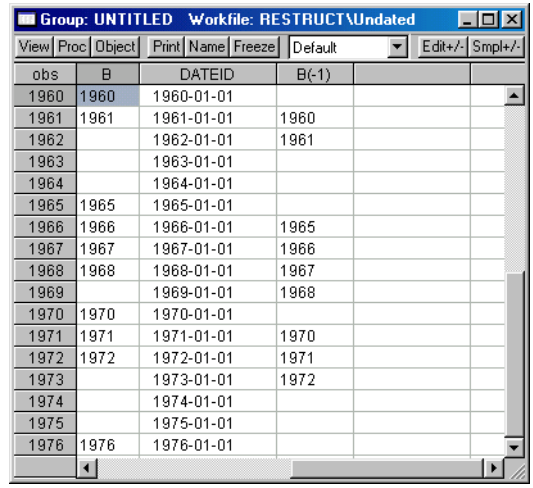


Alternately, we could have chosen to restructure with the **Insert empty obs** checkbox selected, thus ensuring that we have a regular frequency workfile.

To see the effect of this option, we may reopen the **Workfile structure** dialog by double clicking on the "Range:" string near the top of the workfile window, selecting the **Insert empty obs** option, and then clicking on **OK**. EViews will inform

us that the restructure option involves creating 7 additional observations, and will prompt us to continue. Click on **OK** again to proceed. The resulting workfile window will show the additional observations.

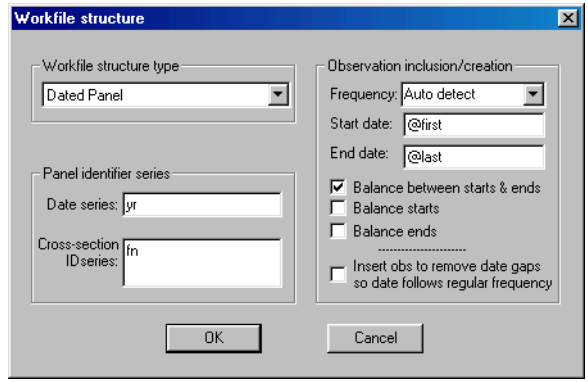
We again show the group containing B, DATEID, and B(-1). Notice that while the observation identifiers and DATEID now include values for the previously missing dates, B and B(-1), do not. When EViews adds observations in the restructure operation, it sets all ordinary series values to NA or missing for those new observations. You are responsible for filling in values as desired.



obs	B	DATEID	B(-1)
1960	1960	1960-01-01	
1961	1961	1961-01-01	1960
1962		1962-01-01	1961
1963		1963-01-01	
1964		1964-01-01	
1965	1965	1965-01-01	
1966	1966	1966-01-01	1965
1967	1967	1967-01-01	1966
1968	1968	1968-01-01	1967
1969		1969-01-01	1968
1970	1970	1970-01-01	
1971	1971	1971-01-01	1970
1972	1972	1972-01-01	1971
1973		1973-01-01	1972
1974		1974-01-01	
1975		1975-01-01	
1976	1976	1976-01-01	

Dated Panels

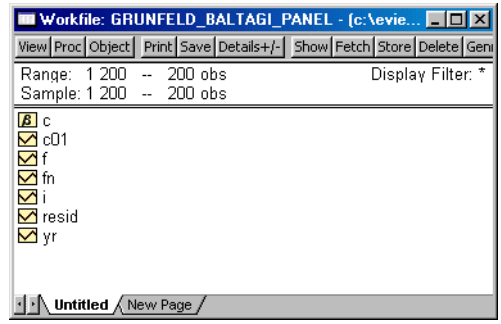
To create a dated panel workfile, you should call up the **Workfile structure** dialog then select **Dated Panel** as our structure type.



There are three parts to the specification of a dated panel. First, you must specify one or more **Date series** that describe date identifiers that are unique within each group. Next, you must specify the **Cross-section ID series** that identify members of a given group. Lastly, you should set options which govern the choice of frequency of your dated data, starting and ending dates, and the adding of observations for balancing the panel or ensuring a regular frequency.

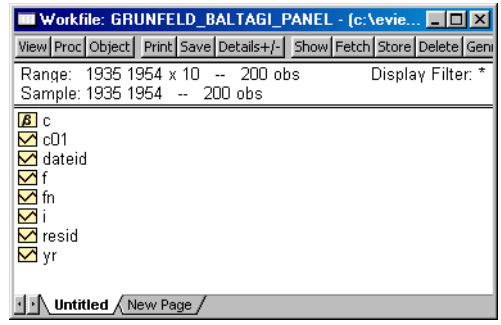
Dated Panel Basics

We begin by considering the Grunfeld data that have been described in a number of places (see, for example, Baltagi, *The Econometric Analysis of Panel Data*, from which this version of the data has been taken). The data measure R&D expenditure and other economic measures for 10 firms for the years 1935 to 1954. These 200 observations form a balanced panel dataset. We begin by reading the data into an unstructured, 200 observation workfile.



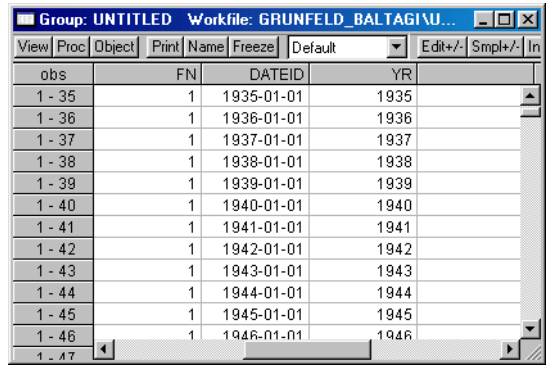
To structure the panel for these data, we call up the **Workfile structure** dialog, select **Dated Panel** as our structure type, and enter the name of the **Cross-section ID series** representing firm number, FN, along with the **Date series** (cell ID) representing the year, YR. If we leave the remaining settings at their default values, EViews will auto detect the frequency of the panel, setting the start and end dates on the basis of the values in the YR series, and will add any observations necessary so that the data between the starts and ends is balanced.

When you click on **OK** to accept these settings, EViews creates a DATEID series, sorts the data by ID and DATEID, locks the two series, and applies the structure. The auto detecting of the date frequency and endpoints yields an annual (balanced) panel beginning in 1935 and ending in 1954.



The basic information about this structure is displayed at the top of the workfile window. There are a total of 200 observations representing a balanced panel of 10 cross-sections with data from 1935 to 1954.

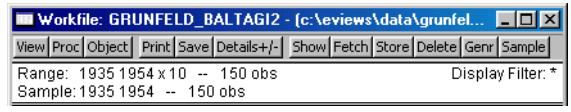
Notice that the observation labels for the structured panel workfile show both the group identifier and the cell identifier.



Dated Panel Balancing

In the basic Grunfeld example, the data originally formed a balanced panel so the various balance operations have no effect on the resulting workfile. Similarly, the option to insert observations to remove gaps has no effect since the data already follow a regular (annual) frequency with no gaps.

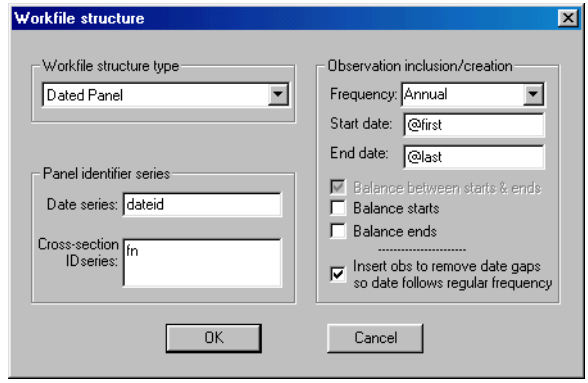
Let us now consider a slightly more complicated example involving panel data that are both unbalanced and irregular. For simplicity, we have created an unbalanced dataset by taking a 150 observation subset of the 200 observations in the Grunfeld dataset.


Workfile: GRUNFELD_BALTAGI2 - (c:\eviews\data\grunfel...
View Proc Object Print Save Details+/- Show Fetch Store Delete Genr Sample
Range: 1935 1954 x 10 -- 150 obs Display Filter: *
Sample: 1935 1954 -- 150 obs

First, we call up the **Workfile structure** dialog and again select **Dated Panel**. We begin by using FN and YR as our group and cell IDs, respectively. Use **Auto detect** to determine the frequency, do not perform any balancing, and click on **OK**. With these settings, our workfile will be structured as an unbalanced, irregular, annual workfile ranging from 1935 to 1954.

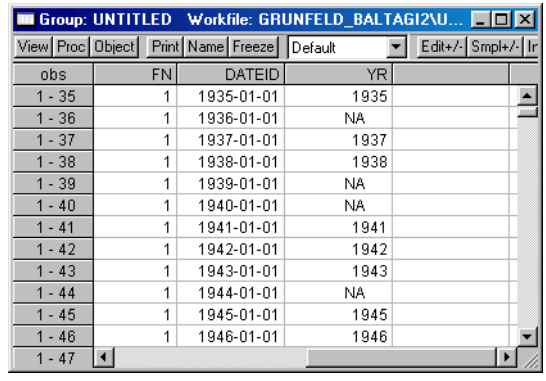
Alternatively, we can elect to perform one or more forms of balancing either at the time the panel structure is put into place, or in a restructure step. Simply call up the **Workfile structure** dialog and select the desired forms of balancing. If you have previously structured your workfile, the dialog will be pre-filled with the existing identifiers and frequency. In this example, we will have our existing annual panel structure with identifiers DATEID and FN.

In addition to choosing whether to **Balance starts** and **Balance ends**, you may choose, at most, one of the two options **Balance between starts and ends**, and **Insert obs to remove date gaps so date follows regular frequency**.



If balancing between starts and ends, the balancing procedure will use the observed cell IDs (in this case, the years encoded in DATEID for all cross-sections) between a given start and end date. All cross-sections will share the same possibly irregular calendar for observations between their starts and ends. If you also elect to insert observations to remove date gaps, EViews balances each cross-section between starts and ends using every date in the calendar for the given frequency. In the latter case, all cross-sections share the same regular calendar for observations between their starts and ends.

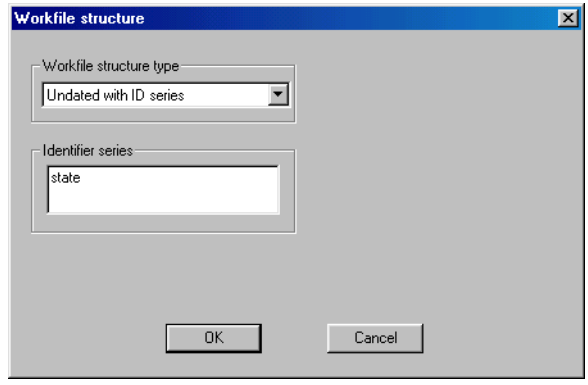
Selecting all three options, **Balance starts**, **Balance ends** and **Balance between starts and ends,** ensures a balanced panel workfile. If we substitute the option **Insert obs to remove date gaps so date follows regular frequency** for **Balance between starts and ends**, we further guarantee that the data follow a regular frequency.



obs	FN	DATEID	YR
1 - 35	1	1935-01-01	1935
1 - 36	1	1936-01-01	NA
1 - 37	1	1937-01-01	1937
1 - 38	1	1938-01-01	1938
1 - 39	1	1939-01-01	NA
1 - 40	1	1940-01-01	NA
1 - 41	1	1941-01-01	1941
1 - 42	1	1942-01-01	1942
1 - 43	1	1943-01-01	1943
1 - 44	1	1944-01-01	NA
1 - 45	1	1945-01-01	1945
1 - 46	1	1946-01-01	1946
1 - 47			

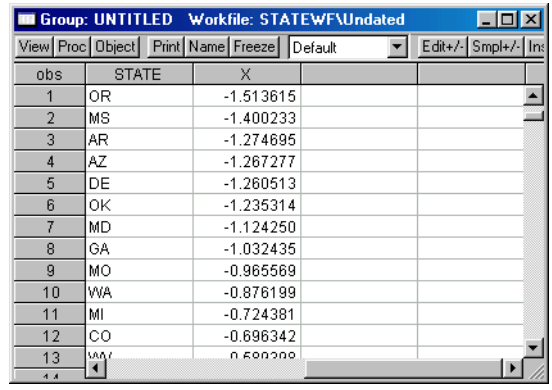
In partly or fully balancing the panel workfile, EViews will add observations as necessary, and update the corresponding data in the identifier series. All other variables will have their values for these observations set to NA. Here, we see that EViews has added data for the two identifier series FN and DATEID while the ordinary series YR values associated with the added observations are missing.

Undated with ID series



If you wish to provide cross-section identifiers for your undated data, select **Undated with identifier series** in the combo box.

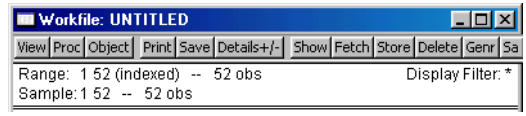
EViews will prompt you to enter the names of one or more ID series. When you click on **OK**, EViews will first sort the workfile by the values of the ID series, and then lock the series so that it may not be altered so long as the structure is in place. The values of the ID series will now be used in place of the default integer identifiers.



Let us consider a simple example. Suppose that we have a 52 observation unstructured workfile, with observations representing the 50 states in the U.S., D.C., and Puerto Rico.

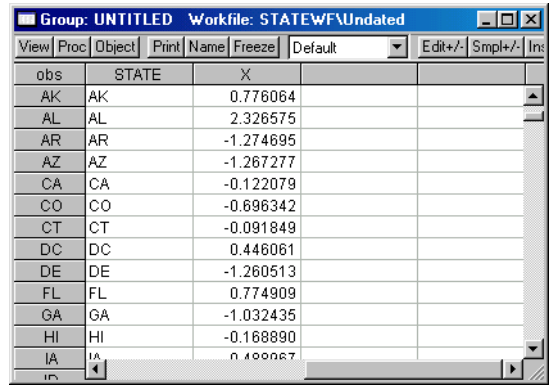
We wish to use the values in the alpha series STATE (which contains the standard U.S. Postal Service abbreviations) to identify the observations. The data for STATE and a second series, X, are displayed here. Notice that the data are ordered from low to high values for X.

Simply select **Undated with identifier series**, enter "state" as the identifier series, and click **OK** to accept the settings. EViews will sort the observations in the workfile by the values in the ID series, and then apply the requested structure, using and locking down the contents of STATE.


Workfile: UNTITLED
View Proc Object Print Save Details+/- Show Fetch Store Delete Genr Sa
Range: 1 52 (indexed) -- 52 obs
Display Filter: *
Sample: 1 52 -- 52 obs

Visually, the workfile window will change slightly with the addition of the description "(indexed)" to the upper portion of the window, showing that the workfile has been structured. Note, however, that since the dataset is still undated, the workfile range and sample are still expressed in integers ("1 52").

To see the two primary effects of structuring cross-section workfiles, we again examine the values of STATE and the variable X. Notice that the data have been sorted (in ascending order) by the value of STATE and that the observation identifiers in the left-hand border now use the values of STATE.

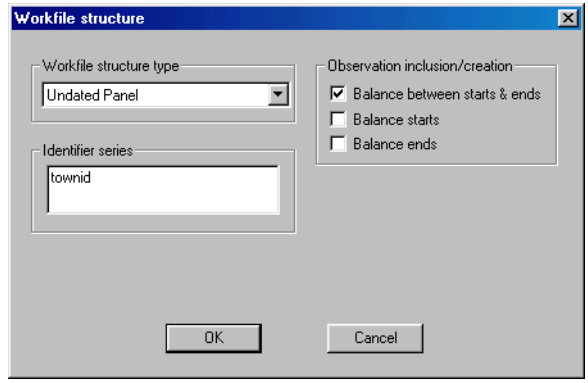


Note that as with irregular structured workfiles, the observation labels will adopt the characteristics of the classifier series display format. If you wish to change the appearance of the observation labels, you should set the spreadsheet display format for STATE (see "Changing the Spreadsheet Display" on page 88).

Undated Panels

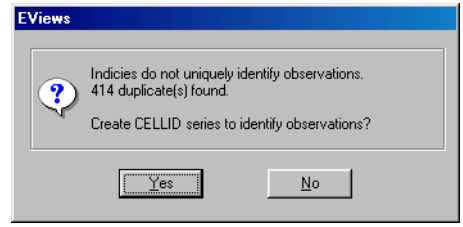
To apply an undated panel structure to your workfile, you must specify one or more **Cross-section ID series** that identify members of a given group. First, select Undated Panel from the combo box, and then enter the names of your **identifier series**. You may optionally instruct EViews to balance between the starts and ends, the starts, or the ends of your data.

As an example, we consider the Harrison and Rubinfeld data on house prices for 506 observations located in 92 towns and cities in the harbor area near New Bedford, MA (Harrison and Rubinfeld 1978; Gilley and Pace 1996).



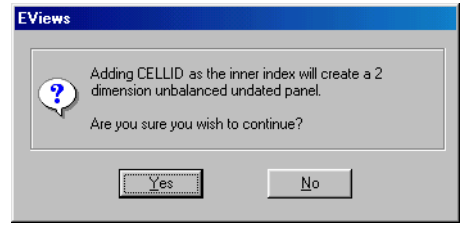
The group identifiers for these data are given by the series TOWNID, in which the town for a given observation is coded from 1 to 92. Observations within a town are not further identified, so there is no cell ID within the data. Here we specify only the group identifier TOWNID.

When we click on **OK**, EViews analyzes the data in TOWNID and determines that there are duplicate observations—there are, for example, 22 observations with a TOWNID of 5. Since TOWNID does not uniquely identify the individual observations, EViews prompts you to create a new cell ID series.



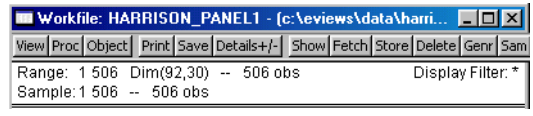
If you click on **No**, EViews will return you to the specification page where you may define a different set of group identifiers. If you choose to continue, EViews will create a new series with a name of the form CELLID## (*e.g.*— CELLID, CELLID01, CELLID02, etc.) containing the default integer cell identifiers. This series will automatically be used in defining the workfile structure.

There are important differences between the two approaches (*i.e.*, creating a new ID series, or providing a second ID series in the dialog) that are discussed in "Lags, Leads, and Panel Structured Data" on page 212. In most circumstances, however, you will click on **Yes** to continue. At this point, EViews will inform you that you have chosen to define a two-dimensional, undated panel, and will prompt you to continue. In this example, the data are unbalanced, which is also noted in the prompt.


EViews
Adding CELLID as the inner index will create a 2 dimension unbalanced undated panel.
Are you sure you wish to continue?
Yes
No

When you click on **Yes** to continue, EViews will restructure the workfile using the identifiers TOWNID and CELLID##. The data will be sorted by the two identifiers, and the two-dimensional panel structure applied. The workfile window will change to show this restructuring. As depicted in the upper portion, we have a 506 observation, undated panel with dimension (92, 30)—92 groups with a maximum of 30 observations in any group.

Note that in this example, balancing the starts or interiors has no effect on the workfile since CELLID## has cell IDs that begin at 1 and run consecutively for every group. If, however, we choose to balance the ends, which vary between 1 and 30, the corresponding resize operation would add 2254 observations. The final result would be a workfile with 2760 observations, comprised of 92 groups, each with 30 observations.


Workfile: HARRISON_PANEL1 - (c:\eviews\data\harri...
View Proc Object Print Save Details+/- Show Fetch Store Delete Genr Sam
Range: 1 506 Dim(92,30) -- 506 obs
Display Filter: *
Sample: 1 506 -- 506 obs

Common Structuring Errors

In most settings, you should find that the process of structuring your workfile is relatively straightforward. It is possible, however, to provide EViews with identifier information that contains errors so that it is inconsistent with the desired workfile structure. In these cases, EViews will either error, or issue a warning and offer a possible solution. Some common errors warrant additional discussion.

Non-unique identifiers

The most important characteristic of observation identifiers is that they uniquely identify every observation. If you attempt to structure a workfile with identifiers that are not unique, EViews will warn you of this fact, will offer to create a new cell ID, and will prompt you to proceed. If you choose to proceed, EViews will then prompt you to create a panel workfile structure using both the originally specified ID(s) and the new cell ID to identify the observations. We have seen an example of this behavior in our discussion of the undated panel workfile type ("Undated Panels" on page 228).

In some cases, however, this behavior is not desired. If EViews reports that your date IDs are not unique, you might choose to go back and either modify or correct the original ID values, or specify an alternate frequency in which the identifiers are unique. For example, the date string identifier values "1/1/2002" and "2/1/2002" are not unique in a quarterly workfile, but are unique in a monthly workfile.

Invalid date identifiers

When defining dated workfile structures, EViews requires that you enter the name or names of series containing date information. This date information may be in the form of valid EViews date values, or it may be provided in numbers or strings which EViews will attempt to interpret as valid date values. In the latter case, EViews will attempt to create a new series containing the interpreted date values.

If EViews is unable to translate your date information into date values, it will issue an error indicating that the date series has invalid values or that it is unable to interpret your date specification. You must either edit your date series, or structure your workfile as an undated workfile with an ID series.

In cases where your date information is valid, but contains values that correspond to unlikely dates, EViews will inform you of this fact and prompt you to continue. Suppose, for example, that you have a series that contains 4-digit year identifiers ("1981", "1982", etc.), but also has one value that is coded as a 2-digit year ("80"). If you attempt to use this series as your date series, EViews will warn you that it appears to be an integer series and will ask you if you wish to recode the data as integer dates. If your proceed, EViews will *alter the values in your series* and create an *integer dated* (*i.e.*, not time dated) workfile, which may not be what you anticipated.

Alternately, you may cancel the restructure procedure, edit your date info series so that it contains valid values, and reattempt to apply a structure.

Missing value identifiers

Your identifier series may be numeric or alpha series containing missing values. How EViews handles these missing values depends on whether the series is used as a date ID series, or as an observation or group ID series.

Missing values are not allowed in situations where EViews expects date information. If EViews encounters missing values in a date ID series, it will issue a warning and will prompt you to delete the corresponding observations. If you proceed, EViews will remove the observations from the workfile. If removed, the observations may not be recovered, even if you subsequently change or remove the workfile structure.

If the missing values are observed in an observation or group ID series, EViews will offer you a choice of whether to keep or remove the corresponding observations, or whether to cancel the restructure. If you choose to keep the observations, the missing value, NA, for numeric series, and a blank string for alpha series, will be used as an observation or cross-section ID in the restructured workfile. If you choose to drop the observations, EViews will simply remove them from the workfile. These observations may not be recovered.

Removing a Workfile Structure

You may remove a workfile structure at any time by restructuring to an unstructured or regular frequency dated workfile. Call up the **Workfile structure** dialog and select **Unstructured/Undated** or **Dated - regular frequency** from the combo box. Fill out the appropriate entries and click **OK**.

EViews will remove the workfile structure and will unlock any series used as date, group, or observation identifiers.

Resizing a Workfile

Resizing a workfile page is a special case of restructuring. Simply call up the **Workfile structure** dialog for any workfile page by selecting **Proc/"Structure/Resize Current Page..."** from a workfile window, or by clicking on the "Range:" description header near the top of the main workfile window. EViews will open the workfile structure dialog with your current settings displayed in the appropriate fields.

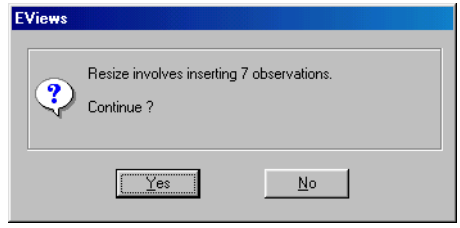
Dated - regular frequency / Unstructured

For workfile types where the structure of the data is described explicitly (dated with regular frequency, or unstructured), the **Start date** and **End date**, or **Observations** values will be filled out with actual values.

To change the sizes of regular frequency workfiles, enter the appropriate **Start date** and **End date** information using explicit dates or offsets from "@FIRST" and "@LAST".

To change the size of an unstructured workfile, change the number of **Observations**. Note that for unstructured data, you may only add or delete observations from the end of the workfile, you may not change the starting observation; if you wish to modify the starting observation you will need to work with an integer dated workfile.

EViews will inform you of the number of observations to be added and/or deleted, and will prompt you to continue.



For example, changing the **End date** for your annual workfile from "2001" to "2009", or the number of **Observations** in your unstructured workfile from "100" to "107" will both add 7 observations to the end of the respective workfiles. Likewise, changing the **Start date** of your monthly workfile from "1990:01" to "@FIRST-24" will add 24 months to the beginning of the workfile while changing the **End date** to "@LAST-3" removes (deletes) the last three observations.

Dated - specified by date series

For a dated workfile that is structured using a date series, the dialog will open with pre-filled **Start date** and **End date** values containing "@FIRST" and "@LAST" as stand-ins for the earliest and latest observed dates. To change the size of a dated workfile structured by a date series, simply enter the appropriate information using explicit dates or offsets from "@FIRST" and "@LAST".

Given your start and end date values, EViews will analyze your date identifiers to determine whether you need to add or remove observations. If required, EViews will inform you of the number of observations to be added or deleted, and you will be prompted to continue. If observations are added, the date series will be modified to hold the corresponding date values. As with other forms of restructuring, deleted observations may not be recovered.

An observation will be deleted if the corresponding date ID falls outside the range implied by the start and end dates. If we enter "1970" as the **Start date** and "2010" as the **End date** in our annual workfile, any observations whose date series value is earlier than 1970 or later than 2010 will be removed from the workfile. If we enter "@FIRST + 2" and "@LAST-3" as our **Start date** and **End date**, EViews will delete the first two and last three observations from the workfile.

EViews will add observations to the workfile if the **Start date** is earlier than "@FIRST" or the **End date** is later than "@LAST". The observations to be added are determined by

examining the regular frequency calendar to find all possible dates which fall in the desired range. If, in our annual workfile that ranges from 1980 to 2000, we specify a **Start date** of "1975", EViews will add observations for all of the years from 1975 to 1979, and will modify the date series so that it contains the associated date values. Alternatively, entering "@FIRST-2" and "@LAST + 2" adds two observations corresponding to 1978 and 1979, and two observations corresponding to 2001 and 2002.

Note that there is a bit of asymmetry here in the use of offsets to "@FIRST" and "@LAST". Offsets that remove observations from the workfile simply count from the first or last observation, while offsets that add observations to the workfile use the regular frequency calendar to determine the dates to be added.

Dated Panel

For dated panel workfiles, the prefilled **Start date** and **End date** values will contain "@FIRST" and "@LAST" as stand-ins for the cross-section specific earliest and latest observed dates. To resize a dated panel workfile, you may enter an explicit date value in one or both of those fields. If you elect to use offsets, you must take care to understand the inherent complexities involved.

When you enter "@FIRST + 2" and "@LAST-2", EViews trims off 2 *observations* from the beginning and end of each cross-section. Used in this fashion, "@FIRST" refers to the earliest date for each cross-section, and the offsets are in observation space.

If we combine this trimming with balancing starts or ends, balancing occurs prior to the trimming of observations. Interestingly, this means that the starts or ends will not necessarily be balanced following trimming.

In order to use "@FIRST-2" or "@LAST + 2", EViews must balance starts or ends. The interpretation of the offsets that extend beyond the range of observations differs since they are evaluated in regular date space. If you enter "@FIRST-2" and choose to balance starts, the behavior is: first balance starts, then add two observations to the beginning in date space. Note that this operation is the same as adding two observations in regular date space to the cross-section with the earliest observed date and then balancing starts.

This behavior means that you cannot easily add two observations (in date space) to the start or end of each cross-section, without possibly adding more via start or end balancing. The panel data will have balanced starts or ends following the operation.

Undated with ID series / Undated Panel

Resizing an undated workfile that is structured using an ID series requires several distinct operations, since there is no simple way to describe the restructure operation. At a deep level, resizing these types of workfiles involves modifying your identifiers, and then adding or deleting observations with specific identifier values.

To alter the identifier series you must first remove the workfile structure. Call up the **Workfile structure** dialog and select **Unstructured/Undated** from the combo box. Click on **OK**. EViews will remove the existing workfile structure and will unlock the ID series.

If you wish to remove observations, you should edit one of the ID series so that the desired observations have missing IDs. If you reapply the original **Undated with ID series** or **Undated Panel** structure, EViews will prompt you to remove observations with the missing ID values. We remind you that this step will remove all observations with missing values for the identifiers; if you originally used the missing value as a valid identifier, the corresponding observation will also be removed.

To add observations, you must first append observations to the workfile by expanding the unstructured workfile and then editing the ID series to add unique identifiers for the new values, or by using the built-in tools to append to the workfile page ("Appending to a Workfile" on page 234). Once you have added the new observations, you may reapply the workfile structure. EViews will sort your data using the identifier values, lock down the ID series, and then apply the structure to the expanded workfile.

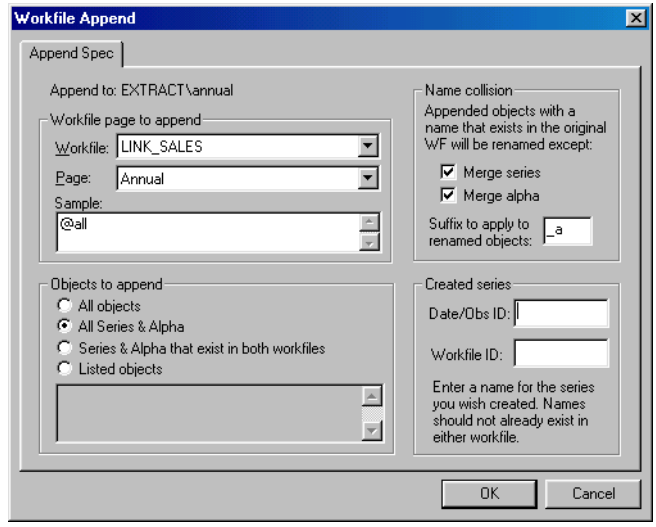
Appending to a Workfile

One method of combining two workfile pages is to append observations from a source workfile page to the end of a target workfile page. When appending data, EViews first removes any structure from the target page, then expands its range to encompass the combined range of the original page and the appended data. The data from the source page are then copied to the expanded part of the target workfile range, either in existing series or alpha objects, or in newly created objects.

When appending, you should first make certain that the workfiles containing both the source and target page are open in EViews. In some cases (for example, concatenating a workfile page with itself), you only need to have a single open workfile since the source and target workfiles are the same.

To open the **Workfile Append** dialog, click on the **Proc** button on the target workfile toolbar and select **Append to Current Page...**, or select **Proc/Append to Current Page...** from the main menu.

Selecting Data to Append



You should begin by selecting a workfile page containing data to be appended to the target page. The first combo box contains a list of all workfiles currently in memory from which you should select the source workfile; in the second combo box, you should choose a page from those in the workfile you have selected. Here, we have instructed EViews to append data from the ANNUAL page in the workfile LINK_SALES.

Next, you should specify a sample of observations in the source page to be appended; any valid EViews sample may be provided. Here, we have specified the default sample "@ALL", which ensures that we use all of the observations in the source page.

If you wish, you may use the **Objects to append** settings to specify the objects to be appended or copied. By default (**All series & alpha**), EViews will append all series and alphas (and links) from the source page into the destination page. If you select **All objects,** EViews will append all series and alphas, and will copy all other objects into the destination. Alternatively, choosing **Listed objects** allows you to specify the individual objects to be copied by name, using wildcards if convenient. To append only those data objects that exist in both pages, you should select **Series & alpha that exist in both workfiles**. If this setting is selected, a series or numeric link Y in the source page will only be appended if a series Y exists in the active page, and an alpha or alpha link X in the source will only be appended if an alpha series X exists in the destination.

Handling Name Collision

The settings in **Name collision** control the method EViews uses to append data when a source object name is present in the target page. To understand the effects of the various settings, consider the three possible scenarios that may occur when appending from an object into a workfile page:

- there is no object with the same name in the target page.

- an object with the same name exists in the target, but the object type is not compatible.
- an object with the same name exists in the target, and the object type is compatible with the source object.

In saying that the source and destination objects are compatible, we indicate that the source data may be added to the end of the existing object. Series and numeric link data may only be added to the end of series objects, while alpha and alpha link data may only be added to the end of alpha objects. All other combinations of objects are said to be incompatible.

Suppose that we wish to append the source series X or numeric link to the target page. If there is no object with the same name in the target page, EViews will create a new series, X, containing NA values for the original target page observations, and the values of the source series X for observations in the expanded part of the range.

If there is an incompatible matching object, a new object will be created with a name formed from the original name and the text specified in the **Suffix to apply to renamed objects** edit field. If, for example, the target page contains an incompatible X (*e.g.*, it contains the equation X), EViews will create a new series using the original name, and the specified suffix, for example, "X_A" (using the default suffix, "_A").

If there is a compatible matching object, EViews will examine your dialog settings to determine the appropriate behavior. By default, EViews will append the data from a compatible source object to the end of the existing object. Thus, data from the series or numeric link X will be copied to the expanded part of the range of the target series X, and data from the alpha or alpha link Y will be copied to the end of the alpha series Y. You may override this default so that EViews creates a new object even when the matching objects are compatible, by unselecting the **Merge series** or **Merge alpha** checkboxes.

Creating Identifier Series

The optional **Created series** settings in the dialog allow you to save series containing information about each observation in the combined workfile.

To save a series containing the date or observation ID associated with each observation in the combined workfile, you should enter a unique name in the edit field labeled **Date/Obs ID**. The specified series will be created in the target page, and will contain the observation or cell identifiers given by the structures associated with the source and the original target pages. Saving the IDs is particularly useful since appending to a workfile removes the existing page structure.

The optional **Workfile ID** series identifies the source of the observation in the combined workfile: observations in the original target page are assigned the value 0, while observations in the appended portion of the target will be given the value 1.

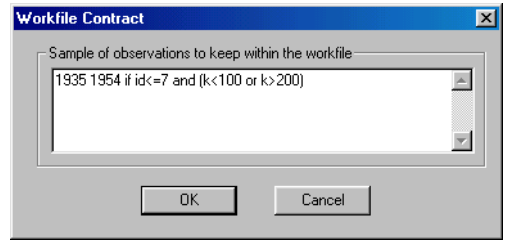
Contracting a Workfile

Samples are an important tool for restricting the set of observations used by EViews when performing calculations. You may, for example, set an estimation sample to restrict the observations used in a regression to only include females, or to only use the observations with dates between 1990 and 2003. An important advantage to working with samples is that the exclusion of observations is temporary, and may be reset simply by providing a new sample specification. Note also that even as they are excluded from calculations, out-of-sample observations still exist, and are used for lag processing.

There may be times, however, when you wish to drop or remove observations from a workfile page. For example, if you have daily data on stock trades and want lags to skip holidays, you must permanently remove holidays from the workfile. Similarly, if the focus of your analysis is on female labor force participation, you may wish to subset your workfile by excluding all males. Contracting the workfile in this fashion both reduces the size of the workfile and makes it easier to work with, since you no longer have to remember to set all samples to exclude males.

To contract a workfile page in place, you should click on the **Proc** button on the workfile toolbar and select **Contract Current Page...,** or select **Proc/Contract Current Page...** from the main menu.

EViews will open the **Workfile Contract** dialog prompting you to input a valid sample specification. Simply enter a sample specification and EViews will drop all observations in the current page that do not meet the specified criteria. Here, we drop all observations where the ID series is greater than 7 or where K lies between 100 and 200 (inclusive).



We emphasize that the workfile contraction occurs in place so that the existing workfile page will no longer exist. If you wish to keep the original page, you should make a copy of the page, or save it to disk.

Copying from a Workfile

EViews provides you with convenient tools for copying or extracting subsamples of observations and series objects from existing workfiles and creating new pages containing the extracted data or links to the data. You may, for example, wish to create separate workfile pages for the males and females in your cross-section workfiles, or to keep only non-holiday dates from your regular frequency daily-7 data. Similarly, you may wish to create a page containing a small subset of the series found in your original workfile.

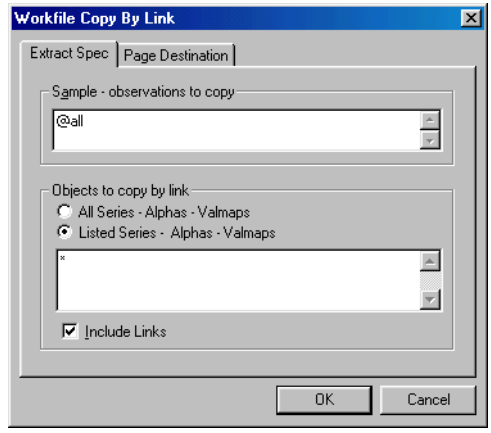
Copying or extracting the series object data may be performed in two distinct ways: by creating links in a new page in the same workfile, or by copying the series objects into a new page in the existing or an alternate workfile.

The first method uses link objects to create memory efficient, dynamically updating copies of the data in your series, link, and alpha objects, but requires that the new destination page be in the same workfile.

The second method copies the actual values in the objects. Since links are not involved, you may use this approach to copy data into new pages in different workfiles. In addition, when copying by value, you may copy other types of EViews objects and you will have access to built-in tools for creating random samples of the observations in the source workfile.

Copying by Link

To copy all or part of the data in a workfile by creating links, you should select **Proc/"Copy Extract from Current Page"/By Link to New Page...**. EViews will open the **Workfile Copy By Link** dialog in which you will specify the data to be copied.



There are two principal ways that you can specify a subset of the data to be copied: you may specify a subsample of observations in the workfile or you may specify a subset of the series objects.

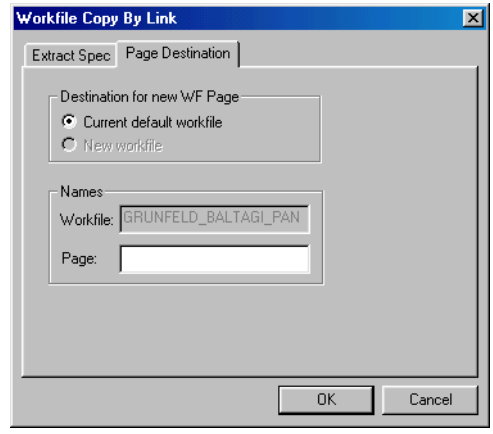
EViews will copy all of the observations in the sample specified in the edit box labeled **Sample - observations to copy**. To specify a subsample of observations, you should replace the default "@ALL" with a valid EViews sample.

You may elect to copy all series, alphas, and valmaps, or you may select the

Listed Series - Alphas - Valmaps radio button and enter a list of the series to be copied, with wildcards, if desired.

If the **Include Links** checkbox is selected, EViews will copy series and alpha links along with ordinary series and alphas. If you uncheck **Include Links**, EViews will drop all link objects from the copy list.

The copy by link procedure will create the links in a new page in the existing workfile. By default, the page will be given a name based on the page structure (e.g., "Annual", or "Daily5"). You may provide a name for this destination page by clicking on the **Page Destination** tab and enter the desired name. If a page with that name already exists in the workfile, EViews will create a new page using the next available name. Note that since we are copying by link, you may not create a page in a different workfile.



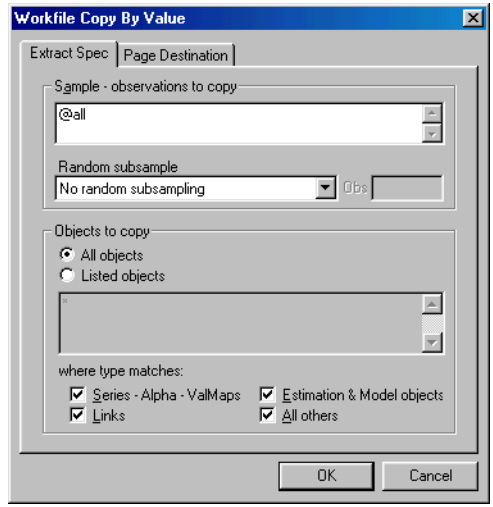
When you click on **OK** to accept the dialog settings, EViews first examines your source workfile and the specified sample, and then creates a new page with the appropriate number of observations.

Next, EViews will copy, by value, the ID series used to structure the source workfile page for the specified sample of observations. Using the new series, EViews will structure the new workfile in a manner similar to the source workfile page. If, for example, you have an undated workfile that is structured using an ID series COUNTRIES, EViews will create a series in the destination page, copy the relevant values, and structure the page as an undated workfile using the new ID series COUNTRIES. Similarly, if the original page has an annual panel structure that is defined using multiple ID series, all of the ID series will be copied to the new page, and the page will be structured as an annual panel using these new series.

Lastly, EViews will create links in the new page for all of the specified series objects. The links will be defined as general match merge links using the source and destination ID series. Since the new page is a subset of the original page, the contraction methods will be set to **No contractions allowed** (see "Link calculation settings" on page 193).

Copying by Value

To copy all or part of the workfile by value, you should select **Proc/"Copy Extract from Current Page"/By Value to New Page or Workfile...**. EViews will open the **Workfile Copy By Value** dialog.



You should first specify an EViews sample describing the observations to be copied. By default, EViews will use the sample "@ALL".

Next, you should use the combo box to select a **Random subsample** method. By default, all of the observations in the sample will be used (**No random subsampling**), but you may choose to extract a random sample in one of three ways:

- You may extract a subsample with a fixed number of observations (**Fixed subsample size - number of observations**). If the specified subsample size is larger than the number of observations, the entire sample is used.
- You may select a subsample with a fixed size, where the number of observations is specified as a percent of the total number of observations (**Fixed subsample size - % of observations**).
- You may take a simple random sample in which every observation has a fixed probability of being selected (**Random subsample size - % applied to each obs**). As the label suggests, the number of observations in the resulting subsample is itself random.

In the remainder of the dialog page you should specify the objects to be copied. There are two parts to the object specification: a list of object names, and a set of modifiers for object types.

By default, the **All objects** radio button is selected so that the list of object names provided to EViews will include every object in the source workfile. You may instead provide an explicit list by clicking on the **Listed objects** radio button and entering the names of objects (using wildcards if appropriate).

The type matching checkboxes (**Series - Alphas - Valmaps**, **Links**, **Estimation & Model Objects**, **All others**) may be used to restrict the object list on the basis of broad classifica-

tions for type; an object will be copied only if it is in the list of object names provided in the edit box, and if its type matches a classification that you elect to copy. If, for example, you wish to remove all objects that are not series objects or valmaps from your list, you should uncheck the **Estimation & Model objects** and the **All others** checkboxes.

Lastly, you may optionally provide a destination workfile page. By default, EViews will copy the data to a new workfile in a page named after the workfile page structure (*e.g.*, "Quarterly", "Monthly"). You may provide an alternative destination by clicking on the **Page Destination** tab in the dialog, and entering the desired destination workfile and/or page.

When you click on **OK,** EViews examines your source workfile and the specified sample, and creates a new page with the appropriate number of observations. EViews then copies the ID series used to structure the source workfile, and structures the new workfile in identical fashion. Lastly, the specified objects are copied to the new workfile page.

Reshaping a Workfile

In a typical study, each subject (individual, firm, period, *etc.*) is observed only once. In these cases, each observation corresponds to a different subject, and each series, alpha, or link in the workfile represents a distinct variable.

In contrast, *repeated measures data* may arise when the same subject is observed at different times or under different settings. The term *repeated measures* comes from the fact that for a given subject we may have repeated values, or measures, for some variables. For example, in longitudinal surveys, subjects may be asked about their economic status on an annual basis over a period of several years. Similarly, in clinical drug trials, individual patient health may be observed after several treatment events.

It is worth noting that standard time series data may be viewed as a special case of repeated measures data, in which there are repeated higher frequency observations for each lower frequency observation. Quarterly data may, for example, be viewed as data in which there are four repeated values for each annual observation. While time series data are not typically viewed in this context, the interpretation suggests that the reshaping tools described in this section are generally applicable to time series data.

There are two basic ways that repeated measures data may be organized in an EViews workfile. To illustrate the different formats, we consider a couple of simple examples.

Suppose that we have the following dataset:

ID1	ID2	Sales
1	Jason	17
1	Adam	8
2	Jason	30
2	Adam	12
3	Jason	20

We may view these data as representing repeated measures on subjects with identifiers given in ID1, or as repeated measures for subjects with names provided in ID2. There are, for example, two repeated values for subjects with "ID1 = 1", and three repeated values for SALES for Jason. Note that in either case, the repeated values for the single series SALES are represented in multiple observations.

We can rearrange the layout of the data into an equivalent form where the values of ID2 are used to break SALES into multiple series (one for each distinct value of ID2):

ID1	SalesJason	SalesAdam
1	17	8
2	30	12
3	20	NA

The series ID2 no longer exists as a distinct series in the new format, but instead appears implicitly in the names associated with the new series (SALESJASON and SALESADAM). The repeated values for SALES are no longer represented by multiple observations, but are instead represented in the multiple series values associated with each value of ID1.

Note also that this representation of the data requires that we add an additional observation corresponding to the case ID1 = 3, ID2 = "Adam". Since the observation did not exist in the original representation, the corresponding value of SALESADAM is set to NA.

Alternatively, we may rearrange the data using the values in ID1 to break SALES into multiple series:

ID2	Sales1	Sales2	Sales3
Jason	17	30	20
Adam	8	12	NA

In this format, the series ID1 no longer exists as a distinct series, but appears implicitly in the series names for SALES1, SALES2, and SALES3. Once again, the repeated responses for

SALES are not represented by multiple observations, but are instead held in multiple series.

The original data format is often referred to as *repeated observations* format, since multiple observations are used to represent the SALES data for an individual ID1 or ID2 value. The latter two representations are said to be in repeated variable or *multivariate* form since they employ multiple series to represent the SALES data.

When data are rearranged so that a single series in the original workfile is broken into multiple series in a new workfile, we term the operation *unstacking the workfile*. Unstacking a workfile converts data from repeated observations to multivariate format.

When data are rearranged so that sets of two or more series in the original workfile are combined to form a single series in a new workfile, we call the operation *stacking the workfile*. Stacking a workfile converts data from multivariate to repeated observations format.

In a time series context, we may have the data in the standard stacked format:

Date	Year	Quarter	Z
2000Q1	2000	1	2.1
2000Q2	2000	2	3.2
2000Q3	2000	3	5.7
2000Q4	2000	4	6.3
2001Q1	2001	1	7.4
2001Q2	2001	2	8.1
2001Q3	2001	3	8.8
2001Q4	2001	4	9.2

where we have added the columns labeled YEAR and QUARTER so that you may more readily see the repeated measures interpretation of the data.

We may rearrange the time series data so that it is unstacked by QUARTER,

Year	Z1	Z2	Z3	Z4
2000	2.1	3.2	5.7	6.3
2001	7.4	8.1	8.8	9.2

or in the alternative form where it is unstacked by YEAR:

Quarter	Z2000	Z2001
1	2.1	7.4
2	3.2	8.1
3	5.7	8.8
4	6.2	9.2

EViews provides you with convenient tools for reshaping workfiles between these different formats. These tools make it easy to prepare a workfile page that is set up for use with built-in pool or panel data features, or to convert data held in one time series representation into an alternative format.

Unstacking a Workfile

Unstacking a workfile involves taking series objects in a workfile page, and in a new workfile, breaking the original series into multiple series.

We employ an *unstacking ID series* in the original workfile to determine the destination series, and an *observation ID series* to determine the destination observation, for every observation in the original workfile. Accordingly, we say that a workfile is "unstacked by" the values of the unstacking ID series.

To ensure that each series observation in the new workfile contains no more than one observation from the existing workfile, we require that the unstacking ID and the observation ID are chosen such that no two observations in the original workfile have the same set of values for the identifier series. In other words, the identifier series must together uniquely identify observations in the original workfile.

While you may use any series in the workfile as your unstacking and observation identifier series, an obvious choice for the identifiers will come from the set of series used to structure the workfile (if available). In a dated panel, for example, the cross-section ID and date ID series uniquely identify the rows of the workfile. We may then choose either of these series as the unstacking ID, and the other as the observation ID.

If we unstack the data by the cross-section ID, we end up with a simple dated workfile with each existing series split into separate series, each corresponding to a distinct cross-section ID value. This is the workfile structure used by the EViews pool object, and is commonly used when the number of cross-sectional units is small. Accordingly, one important application of unstacking a workfile involves taking a page with a panel structure and creating a new page suitable for use with EViews pool objects.

On the other hand, if we unstack the panel workfile by date (using the date ID series or @DATE), we end up with a workfile where each row represents a cross-sectional unit, and

each original series is split into separate series, one for each observed time period. This format is frequently used in the traditional repeated measures setting where a small number of variables in a cross-sectional dataset have been observed at different times.

To this point, we have described the unstacking of panel data. Even if your workfile is structured using a single identifier series, however, it may be possible to unstack the workfile by first splitting the single identifier into two parts, and using the two parts as the identifier series. For example, consider the simple quarterly data given by:

Date	X	Y
2000Q1	NA	-2.3
2000Q2	5.6	-2.3
2000Q3	8.7	-2.3
2000Q4	9.6	-2.3
2001Q1	12.1	1.6
2001Q2	8.6	1.6
2001Q3	14.1	1.6
2001Q4	15.2	1.6

Suppose we wish to unstack the X series. We may split the date identifier into a year component and a quarter component (using, say, the EViews @YEAR and @QUARTER functions). If we extract the QUARTER and YEAR from the date and use the QUARTER as the unstacking identifier, and the YEAR as the observation identifier, we obtain the unstacked data:

Year	X1	X2	X3	X4
2000	NA	5.6	8.7	9.6
2001	12.1	8.6	14.1	15.2

Note that we have chosen to form the series names by concatenating the name of the X series, and the values of the QUARTER series.

Alternatively, if we use YEAR as the unstacking ID, and QUARTER as the observation ID, we have:

Quarter	X2000	X2001
1	NA	12.1
2	5.6	8.6
3	8.7	14.1
4	9.6	15.2

In some cases, a series in the original workfile will not vary by the unstacking ID. In our example, we have a series Y that is only updated once a year. Stacking by QUARTER yields:

Year	Y1	Y2	Y3	Y4
2000	-2.3	-2.3	-2.3	-2.3
2001	1.6	1.6	1.6	1.6

Since there is no change in the observations across quarters, these data may be written as:

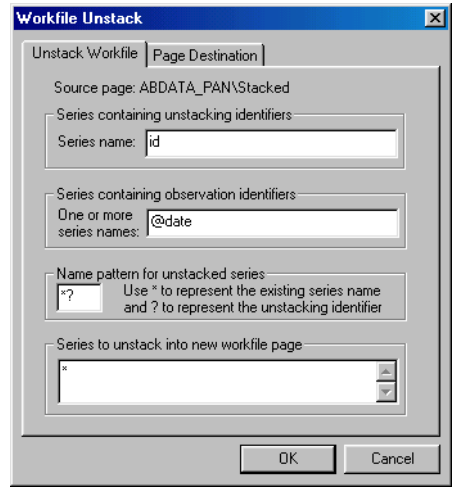
Year	Y
2000	-2.3
2001	1.6

without loss of information. When unstacking, EViews will automatically avoid splitting any series which does not vary across different values of the unstacking ID. Thus, if you ask EViews to unstack the original Y by QUARTER, only the compacted (single series) form will be saved. Note that unstacking by YEAR will *not* produce a compacted format since Y is not constant across values of YEAR for a given value of QUARTER.

Unstacking a Workfile in EViews

To unstack the active workfile page, you should select **Proc/Reshape Current Page/ Unstack in New Page...** from the main workfile menu. EViews will respond by opening the tabbed **Workfile Unstack** dialog.

When unstacking data, there are four key pieces of information that should be provided: a series object that contains the unstacking IDs, a series object that contains the observation IDs, the series in the source workfile that you wish to unstack, and a rule for defining names for the unstacked series.



Unstacking Identifiers

To unstack data contained in a workfile page, your source page must contain a series object containing the unstacking identifiers associated with each observation. For example, you may have an alpha series containing country abbreviations ("US", "JPN", "UK"), or individual names ("Joe Smith", "Jane Doe"), or a numeric series with integer identifiers ("1", "2", "3", "50", "100", ...). Typically, there will be repeated observations for each of the unique unstacking ID values.

You should provide the name of your unstacking ID series object in the top edit field of the dialog. When unstacking, EViews will create a separate series for each distinct value of the ID series, with each of these series containing the multiple observations associated with that value. The series used as the unstacking ID is always dropped from the destination workfile since its values are redundant since they are built into the multiple series names.

If you wish to unstack using values in more than one series, you must create a new series that combines the two identifiers by identifying the subgroups, or you may simply repeat the unstacking operation.

Observation Identifiers

Next, you must specify a series object containing an observation ID series in the second edit field. The values of this series are used to identify both the individual observations in the unstacked series and the structure of the destination page.

Once again, if your workfile is structured, an obvious choice for the unstacking identifier series comes from the series used to structure the workfile, either directly (the date or cross-section ID in a panel page), or indirectly (the YEAR or QUARTER extracted from a quarterly date).

EViews will, if necessary, create a new observation ID series in the unstacked page with the same name as, and containing the unique values of, the original observation ID series. This series will be used to structure the workfile.

If the original observation ID is an ordinary series or alpha, the new page will be structured as a cross-section page using the new identifier series. Alternatively, if the observation ID is a date series or the "@DATE" keyword, EViews will analyze the observed date values and will create a dated page with the appropriate frequency.

Series to be Unstacked

You may enter the names of the series, alphas, and links that you wish to unstack in the edit field **Series to be unstacked into new workfile page.** You may enter the names directly, or use expressions containing wildcards. For example, the expression "SALES A*" instructs EViews to unstack both the SALES series as well as all series objects beginning with the letter "A".

Note that the RESID series and the unstacking ID series may not be unstacked.

Naming Unstacked Series

EViews will use the pattern in the **Name pattern for unstacked series** field to construct the names for the new unstacked series or alphas associated with each stacked series object.

By default, the wildcard pattern "*?" will be used, meaning that unstacked series names will be constructed by concatenating the name of the series object to be unstacked and a string containing one of the unique values found in the unstacking ID series.

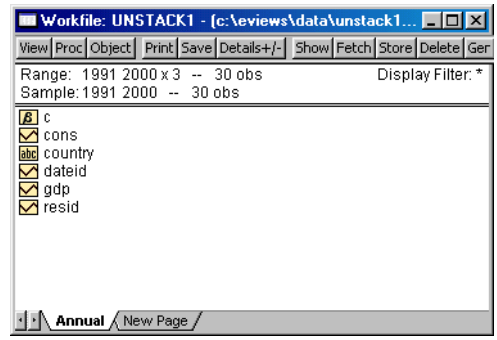
In our example above, when unstacking the SALES series using NAME as the unstacking ID series and the wildcard name pattern "*?", EViews will create the series JASONSALES and ADAMSALES. If instead, we enter the pattern "?_*", EViews will put the unstacked values in the series SALES_JASON and SALES_ADAM.

Unstacking Destination

By default, EViews will unstack the data in a new UNTITLED page in the existing workfile. You may provide an alternative destination by clicking on the **Page Destination** tab in the dialog, and entering the desired destination.

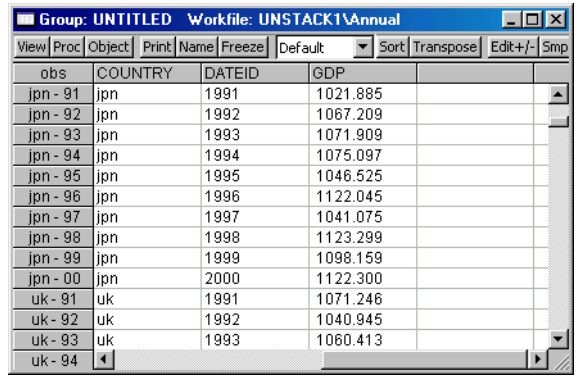
An Example

Consider a workfile that contains the series GDP and CONS, which contain the values of Gross Domestic Product and consumption for three countries stacked on top of each other.



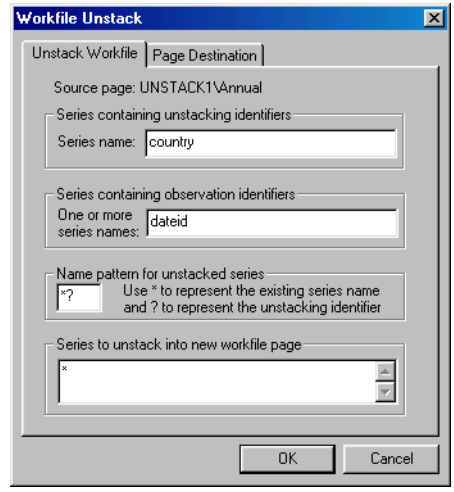
Suppose further that there is an alpha object called COUNTRY containing the values "US", "UK", and "JPN", which identify the country associated with each observation on GDP and CONS. Finally, suppose there is a date series DATEID which identifies the date associated with each observation in the page. COUNTRY and DATEID uniquely determine the observation identifiers.

In our example, we assume that the source page contains annual data from 1991 to 2000 for the three countries in our panel. We can better see this structure by opening a group window showing the values of COUNTRY, DATEID (displayed in year-date format), and GDP.



obs	COUNTRY	DATEID	GDP
jpn - 91	jpn	1991	1021.885
jpn - 92	jpn	1992	1067.209
jpn - 93	jpn	1993	1071.909
jpn - 94	jpn	1994	1075.097
jpn - 95	jpn	1995	1046.525
jpn - 96	jpn	1996	1122.045
jpn - 97	jpn	1997	1041.075
jpn - 98	jpn	1998	1123.299
jpn - 99	jpn	1999	1098.159
jpn - 00	jpn	2000	1122.300
uk - 91	uk	1991	1071.246
uk - 92	uk	1992	1040.945
uk - 93	uk	1993	1060.413
uk - 94			

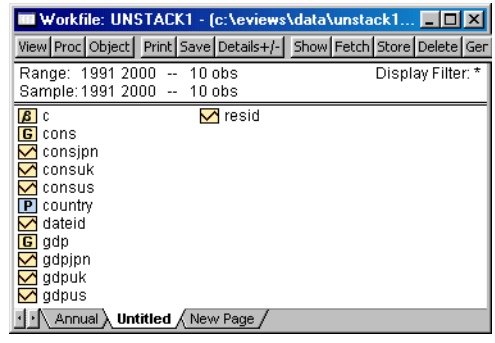
We wish to unstack the data in GDP and CONS using the unstacking ID values in COUNTRY, and the observation IDs in DATEID. Click on **Proc/Reshape Current Page/Unstack in New Page...** in the workfile window to bring up the unstacking dialog.



Enter "COUNTRY" as the unstacking ID series, and "DATEID" for the observation identifier. We leave the remainder of the dialog settings at the default values, so that EViews will use "*?" as the name pattern, will copy all series objects in the page (with the exception of RESID and COUNTRY), and will place the results in a new page in the same workfile.

If you click on **OK** to accept the settings, EViews will first examine the DATEID series to determine the number of unique observation identifiers. Note that the number of unique observation identifier values determines the number of observations in the unstacked workfile. Next, EViews will determine the number of unique values in COUNTRY, which is equal to the number of unstacked series created for each stacked series.

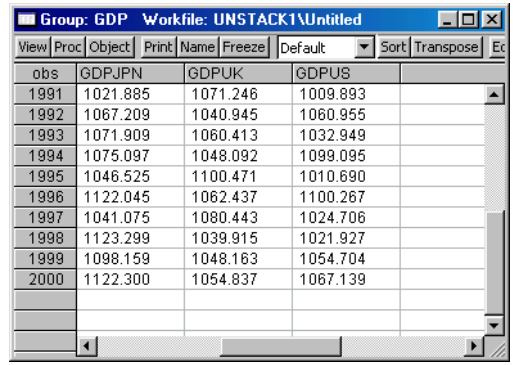
In this example, we start with a balanced panel with 10 distinct values for DATEID, and three distinct values in COUNTRY. The resulting UNTITLED workfile page will follow an annual frequency from the 10 observations from 1991 to 2000, and will have three unstacked series corresponding to each of the source series. The names of these series will be formed by taking the original series name and appending the distinct values in COUNTRY ("US", "UK", and "JPN").



Note that in addition to the six unstacked series CONSJPN, CONSUK, CONSUS, GDPJPN, GDPUK, GDPUS, EViews has created four additional objects. First, the unstacked page contains two group objects taking the name of, and corresponding to, the original series CONS and GDP.

Each group contains all of the unstacked series, providing you with easy access to all of the series associated with the original stacked series. For example, the group GDP contains the three series, GDPJPN, GDPUK, and GDPUS, while CONS contains CONSJPN, CONSUK, and CONSUS.

Opening the GDP group spreadsheet, we see the result of unstacking the original GDP series into three series: GDPJPN, GDPUK, and GDPUS. In particular, the values of the GDPJPN and GDPUK series should be compared with the values of GDP depicted in the group spreadsheet view of the stacked data.



obs	GDPJPN	GDPUK	GDPUS
1991	1021.885	1071.246	1009.893
1992	1067.209	1040.945	1060.955
1993	1071.909	1060.413	1032.949
1994	1075.097	1048.092	1099.095
1995	1046.525	1100.471	1010.690
1996	1122.045	1062.437	1100.267
1997	1041.075	1080.443	1024.706
1998	1123.299	1039.915	1021.927
1999	1098.159	1048.163	1054.704
2000	1122.300	1054.837	1067.139

Second, EViews has created a (date) series DATEID containing the distinct values of the observation ID series. If necessary, this series will be used to structure the unstacked workfile.

Lastly, EViews has created a pool object named COUNTRY, corresponding to the specified unstack ID series, containing all of the unstacking identifiers. Since the unstacked series have names that were created using the specified name pattern, this pool object is perfectly set up for working with the unstacked data.

Stacking a Workfile

Stacking a workfile involves combining sets of series with related names into single series, or repeatedly stacking individual series into single series, and placing the results in a new workfile. The series in a given set to be stacked may be thought of as containing repeated measures data on a given variable. The individual series may be viewed as ordinary, non-repeated measures data.

The stacking operation depends crucially on the set of stacking identifiers. These identifiers are used to determine the sets of series, and the number of times to repeat the values of individual series.

In order for all of the series in a given set to be stacked, they must have names that contain a common component, or *base name*, and the names must differ systematically in containing an identifier. The identifiers can appear as a suffix, prefix, or even in the middle of the base name, but they must be used consistently across all series in each set.

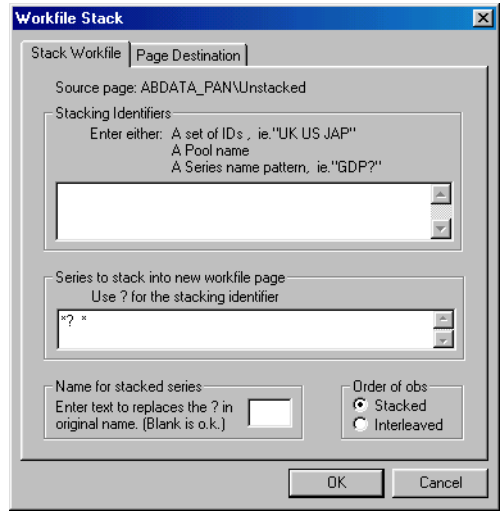
Suppose, for example, we have a workfile containing the individual series Z, and the two groups of series: XUS, XUK and XJPN, and US_Y, UK_Y, and JPN_Y. Note that within each set of series, the identifiers “US”, “UK”, and “JPN” are used, and that they are used consistently within each set of series.

If we employ the set of three identifier values “US”, “UK”, and “JPN” to stack our workfile, EViews will stack the three series XUS, XUK, and XJPN on top of each other, and the series US_Y, UK_Y, and JPN_Y on top of each other. Furthermore, the individual series Z will be

stacked on top of itself three times so that there are three copies of the original data in the new series.

Stacking a Workfile in EViews

To stack the data in an existing workfile page, you should select **Proc/Reshape Current Page/Stack in New Page...** from the main workfile menu. EViews will respond by opening the tabbed **Workfile Stack** dialog.



There are two key pieces of information that you must provide in order to create a stacked workfile: the set of stack ID values, and the series that you wish to stack. This information should be provided in the two large edit fields. The remaining dialog settings involve options that allow you to modify the method used to stack the series and the destination of the stacked series.

Stacking Identifiers

There are three distinct methods that you may use to specify your stack ID values:

First, you may enter a space separated list containing the individual ID values (*e.g.*, "1 2 3", or "US UK JPN"). This is the most straightforward method, but can be cumbersome if you have a large list of values.

Second, you may enter the name of an existing pool object that contains the identifier values.

Lastly, you may instruct EViews to extract the ID values from a set of series representing repeated measures on some variable. To use this method, you should enter a series name pattern containing the base name and the "?" character in place of the IDs. EViews will use this expression to identify a set of series, and will extract the ID values from the series names. For example, if you enter "SALES?", EViews will identify all series in the workfile with names beginning with the string "SALES", and will form a list of identifiers from the remainder of the observed series names. In our example, we have the series SALES1, SALES2, and SALES3 in the workfile, so that the list of IDs will be "1", "2", and "3".

Series to be Stacked

Next, you should enter the list of series, alphas, and links that you wish to stack. Sets of series objects that are to be treated as repeated measures (stacked on top of each other)

should be entered using "?" series name patterns, while individual series (those that should be repeatedly stacked on top of themselves), should be entered using simple names or wildcards.

You may specify the repeated measures series by listing individual stacked series with "?" patterns ("CONS? EARN?"), or you may use expressions containing the wildcard character "*" ("*?" and "?C*") to specify multiple sets of series. For example, entering the expression "?C* ?E*" will tell EViews to find all repeated measures series that begin with the letters "C" or "E" (*e.g.*, "CONS? CAP? EARN? EXPER?"), and then to stack (or interleave) the series using the list of stack ID values. If one of the series associated with a particular stack ID does not exist, the corresponding stacked values will be assigned the value NA.

Individual series may also be stacked. You may list the names of individual simple series (*e.g.*, "POP INC"), or you can specify your series using expressions containing the wildcard character "*" ("*", "*C", "F*"). The individual series will repeatedly be stacked (or interleaved), once for each ID value. If the target workfile page is in the same workfile, EViews will create a link in the new page; otherwise, the stacked series will contain repeated copies of the original values.

It should be noted that the wildcard values for individual series are processed *after* the repeated measures series are evaluated, so that a given series will only be used once. If a series is used as part of a repeated measures series, it will not be used when matching wildcards in the list of individual series to be stacked.

The default value "*? *" is suitable for settings where the repeated series have names formed by taking the base name and appending the stack ID values. The default will stack all repeated measures series, and all remaining individual series (except for RESID). Entering "*" alone will copy or link all series, but does not identify any repeated measures series.

Naming Stacked Series

Stacked individual series will be named in the destination page using the name of the series in the original workfile; stacked repeated measures series will, by default, be named using the base name. For example, if you stack the repeated measures series "SALES?" and the individual series GENDER, the corresponding stacked series will, by default, be named "SALES" and "GENDER", respectively.

This default rule will create naming problems when the base name of a repeated measures series is also the name of an individual series. Accordingly, EViews allows you to specify an alternative rule for naming your stacked repeated measures series in the **Name for stacked series** section of the dialog.

The default naming rule may be viewed as one in which we form names by replacing the "?" in the original specification with a blank space. To replace the "?" with a different

string, you should enter the desired string in the edit field. For example, if you enter the string "_STK", then EViews will name the stacked series "CONS?" and "EARN?" as "CONS_STK" and "EARN_STK" in the destination workfile.

Stacking Order

EViews will, by default, create series in the new page by stacking series on top of one another. If we have identifiers "1", "2", and "3", and the series SALES1, SALES2, and SALES3, EViews will stack the entire series SALES1 followed by the entire series SALES2, followed by SALES3.

You may instruct EViews to interleave the data, by selecting the **Interleaved** radio button in the **Order of Obs** section of the dialog. If selected, EViews will stack the first observations for SALES1, SALES2, and SALES3, on top of the second observations, and so forth.

It is worth pointing out that stacking by series means that the observations contained in a given series will be kept together in the stacked form, while interleaving the data implies that the multiple values for a given original observation will be kept together. In some contexts, one form may be more natural than another.

In the case where we have time series data with different series representing different countries, stacking the data by series means that we have the complete time series for the "US" (USGDP), followed by the time series for the "UK" (UKGDP), and then "JPN" (JPNGDP). This representation is more natural for time series analysis than interleaving so that the observations for the first year are followed by the observations for the second year, and so forth.

Alternatively, where the series represent repeated measures for a given subject, stacking the data by series arranges the data so that all of the first measures are followed by all of the second measures, and so on. In this case, it may be more natural to interleave the data, so that all of the observations for the first individual are followed by all of the observations for the second individual, and so forth.

One interesting case where interleaving may be desirable is when we have data which has been split by period, within the year. For example, we may have four quarters of data for each year:

Year	XQ1	XQ2	XQ3	XQ4
2000	NA	5.6	8.7	9.6
2001	12.1	8.6	14.1	15.2

If we stack the series using the identifier list "Q1 Q2 Q3 Q4", we get the data:

Year	ID01	X
2000	Q1	NA
2001	Q1	12.1
2000	Q2	5.6
2001	Q2	8.6
2000	Q3	8.7
2001	Q3	14.1
2000	Q4	9.6
2001	Q4	15.2

which is not ordered in the traditional time series format from earliest to latest. If instead, we stack by "Q1 Q2 Q3 Q4" but interleave, we obtain the standard format:

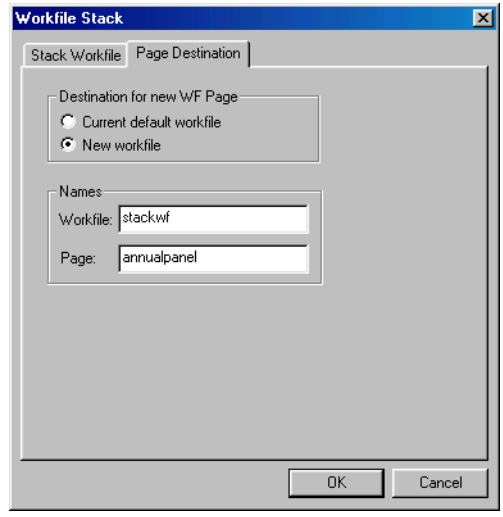
Year	ID01	X
2000	Q1	NA
2000	Q2	5.6
2000	Q3	8.7
2000	Q4	9.6
2001	Q1	12.1
2001	Q2	8.6
2001	Q3	14.1
2001	Q4	15.2

Note that since interleaving changes only the order of the observations in the workfile and not the structure, we can always sort or restructure the workfile at a later date to achieve the same effect.

Stacking Destination

By default, EViews will stack the data in a new page in the existing workfile named "UNTITLED" (or the next available name, "UNTITLED1", "UNTITLED2", *etc.*, if there are existing pages in the workfile with the same name).

You may provide an alternative destination for the stacked data by clicking on the **Page Destination** tab in the dialog, and entering the desired destination.



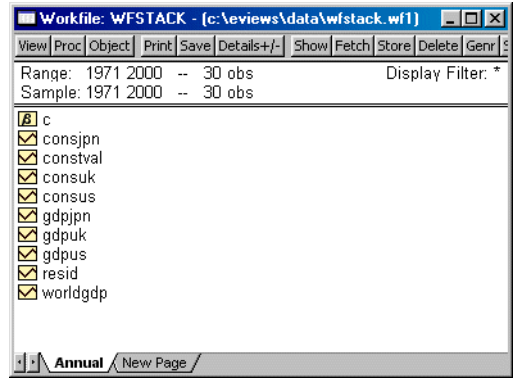
Here, we instruct EViews to put the stacked series in the workfile named STACKWF in the named page ANNUALPANEL. If a page with that name already exists in the workfile, EViews will create a new page using the next available name.

We note that if you are stacking individual series, there is an important consequence of specifying a different workfile as the destination for your stacked series. If the target page is in the same workfile as the original page, EViews will stack individual series by creating link objects in the new page. These link objects have the standard advantages of being memory efficient and dynamically updating. If, however, the target page is in a different workfile, it is not possible to use links, so the stacked series will contain repeated copies of the original individual series values.

An Example

Consider an annual (1971 to 2000) workfile, WFSTACK, that contains the six series: CONSUS, CONSUK, CONSJPN, and GDPUS, GDPUK, GDPJPN, along with the ordinary series CONSTVAL and WORLDGDP.

We wish to stack series in a new page using the stack IDs: “US”, “UK”, and “JPN”.



Click on the **Proc** button and select **Reshape Current Page/Stack in new Page...**.

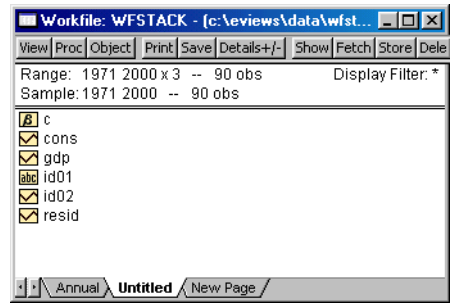
We may specify the stacked series list explicitly by entering “US UK JPN” in the first edit box, or we can instruct EViews to extract the identifiers from series names by entering “GDP?”. Note that we cannot use “CONS?” due to the presence of the series CONSTVAL.

Assuming that we have entered one of the above in **Stacking identifiers** edit box, we may then enter the expression

```
gdp? cons?
```

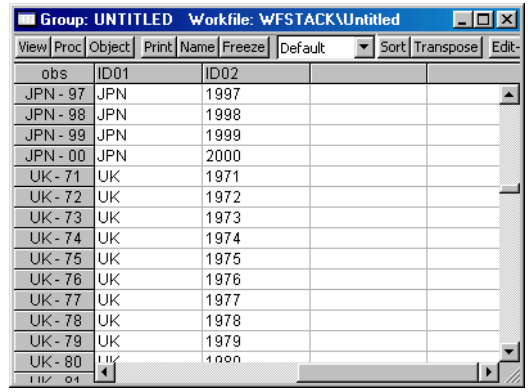
as our **Series to stack**. We leave the remainder of the dialog settings at their defaults, and click on **OK**.

EViews will first create a new page in the existing workfile and then will stack the GDPUS, GDPUK, and GDPJPN series and the CONSUS, CONSUK, and CONSJPN series. Since the dialog settings were retained at the default values, EViews will stack the data by series, with all of the values of GDPUS followed by the values of GDPUK and then the values GDPJPN, and will name the stacked series GDP and CONS.



Here we see the resulting workfile page UNTITLED, containing the stacked series GDP and CONS, as well as two EViews created series objects, ID01 and ID02, that contain identifiers that may be used to structure the workfile.

ID01 is an alpha series that contains the stack ID values "US", "UK", and "JPN" which are used as group identifiers, and ID02 is a data series containing the year observation identifiers (more generally, ID02 will contain the values of the observation identifiers from the original page).



obs	ID01	ID02
JPN - 97	JPN	1997
JPN - 98	JPN	1998
JPN - 99	JPN	1999
JPN - 00	JPN	2000
UK - 71	UK	1971
UK - 72	UK	1972
UK - 73	UK	1973
UK - 74	UK	1974
UK - 75	UK	1975
UK - 76	UK	1976
UK - 77	UK	1977
UK - 78	UK	1978
UK - 79	UK	1979
UK - 80	UK	1980

You may notice that EViews has already applied a panel structure to the page, so that there are three cross-sections of annual data from 1971 to 2000, for a total of 90 observations.

Note that EViews will only apply a panel structure to the new page if we stack the data by series, but not if we interleave observations. Here, had we chosen to interleave, we would obtain a new 90 observation unstructured page containing the series GDP and CONS and the alpha ID01 and series ID02, with the observations for 1971 followed by observations for 1972, and so forth.

We may add our individual series to the stacked series list, either directly by entering their names, or using wildcard expressions. We may use either of the stack series expressions:

```
gdp? cons? worldgdp constval
```

or

```
gdp? cons? *
```

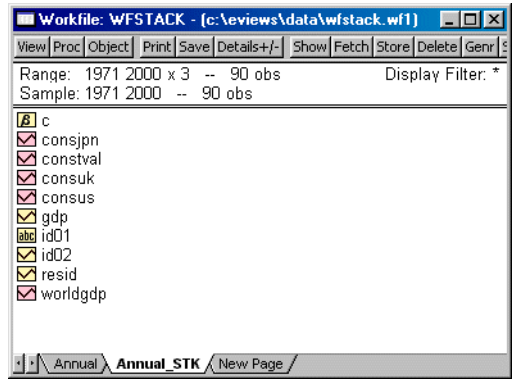
to stack the various "GDP?" and "CONS?" series on top of each other, and the individual series WORLDGDP and CONSTVAL will be linked to the new page so that the original series values are repeatedly be stacked on top of themselves.

It is worth reminding you that the wildcard values for individual series are processed *after* the repeated measures series "GDP?" and "CONS?" are evaluated, so that a given series will only be used once. Thus, in the example above, the series CONSUS is used in forming the stacked CONS series, so that it is ignored when matching the individual series wildcard.

If we had instead entered the list

```
gdp? *
```

EViews would stack the various "GDP?" series on top of each other, and would also link the individual series CONSUS, CONSUK, CONSJPN, WORLDGDP, and CONSTVAL so that the values are stacked on top of themselves. In this latter case, the wildcard implies that since the series CONSUS is not used in forming a stacked repeated measures series, it is to be used as a stacked individual series.



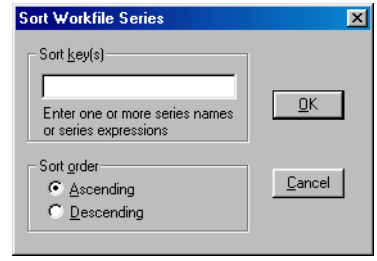
Lastly, we note that since EViews will, by default, create a new page in the existing workfile, all individual series will be stacked or interleaved by creating link objects. If, for example, you enter the stack series list

```
gdp? cons? worldgdp con-
   stval
```

the series WORLDGDP and CONSTVAL will be linked to the destination page using the ID02 values. Alternately, if we were to save the stacked data to a new workfile, by clicking on the **Page Destination** tab and entering appropriate values, EViews will copy the original WORLDGDP and CONSTVAL series to the new page, repeating the values of the original series in the stacked series.

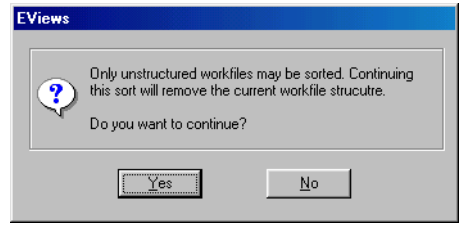
Sorting a Workfile

Basic data in workfiles are held in objects called series. If you click on **Proc/Sort Series...** in the workfile toolbar, you can sort all of the series in an unstructured workfile on the basis of the values of one or more of the series. A dialog box will open where you can provide the details about the sort.



If you list two or more series, EViews uses the values of the second series to resolve ties in the first series, and values of the third series to resolve ties in the first and second, and so forth. If you wish to sort in descending order, select the appropriate option in the dialog.

EViews will only sort unstructured workfiles since sorting a dated or structured workfile will break the link between an observation and the corresponding date identifier.


EViews
Only unstructured workfiles may be sorted. Continuing this sort will remove the current workfile strucutre.
Do you want to continue?
Yes
No

If you attempt to sort a dated or structured workfile, EViews will display a warning informing you that it will first unstructure your data, and then sort the data. Click on **OK** to continue with the operation.

Exporting from a Workfile

MicroTSP Files

You can read or write your workfile in a format that is compatible with MicroTSP. The **Files of type** and **Save as type** combo boxes in the Open and SaveAs dialogs allow you to handle DOS and Macintosh MicroTSP files. Simply click on the combo box and select either **Old Dos Workfile** or **Old Mac Workfile**, as appropriate. You should be aware, however, that if you choose to save a workfile in MicroTSP format, only basic series data will be saved—the remainder of the workfile contents will be discarded.

Foreign Formats

To save your series (and possibly value map data) into a foreign data source, first select **File/Save As...**, from the workfile menu to bring up the standard file **Save** dialog. Clicking on the **Files of type** combo box brings up a list of the output file types that EViews currently supports.

The data export interface is available for Microsoft Access, Aremos TSD, Gauss Dataset, GiveWin/Pc-Give, Rats 4.x, Rats Portable, SAS program files, SAS Transport, native SPSS (using the SPSS Input/output .DLL installed on your system), SPSS Portable, Stata, TSP

Portable, Excel, raw ASCII or binary files, or ODBC Databases (using the ODBC driver already present on your system).

Chapter 10. EViews Databases

An EViews database resembles a workfile in that it is used to contain a collection of EViews objects. It differs from a workfile in two major ways. First, unlike a workfile, the entire database need not be loaded into memory in order to access an object inside it; an object can be fetched or stored directly to or from the database on disk. Second, unlike a workfile page, the objects in a database are not restricted to being of a single frequency or range. A database could contain a collection of annual, monthly, and daily series, all with different numbers of observations.

EViews databases also differ from workfiles in that they support powerful query features which can be used to search through the database to find a particular series or a set of series with a common property. This makes databases ideal for managing large quantities of data.

While EViews has its own native storage format for databases, EViews also allows direct access to data stored in a variety of other formats through the same database interface. You can perform queries, copy objects to and from workfiles and other databases, and rename and delete objects within a database, all without worrying about in what format the data is actually stored.

Database Overview

An EViews database is a set of files containing a collection of EViews objects. In this chapter we describe how to:

- Create a new database or open an existing database.
- Work with objects in the database, including how to store and fetch objects into workfiles, and how to copy, rename and delete objects in the database.
- Use auto-series to work with data directly from the database without creating a copy of the data in the workfile.
- Use the database registry to create shortcuts for long database names and to set up a search path for series names not found in the workfile.
- Perform a query on the database to get a list of objects with particular properties.
- Use object aliases to work with objects whose names are illegal or awkward.
- Maintain a database with operations such as packing, copying, and repairing.
- Work with remote database links to access data from remote sites.

Database Basics

What is an EViews Database?

An EViews native format database consists of a set of files on disk. There is a main file with the extension .EDB which contains the actual object data, and a number of index files with extensions such as .E0, .E1A and .E1B which are used to speed up searching operations on the database. In normal use, EViews manages these files for the user, so there is no need to be aware of this structure. However, if you are copying, moving, renaming, or deleting an EViews database from outside of EViews (using Windows Explorer for example), you should perform the operation on both the main database file and all the index files associated with the database. If you accidentally delete or damage an index file, EViews can regenerate it for you from the main data file using the repair command (see "Maintaining the Database" on page 287).

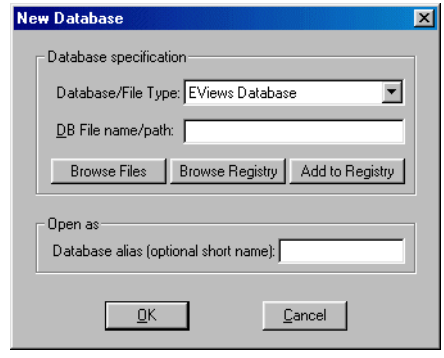
The fact that EViews databases are kept on disk rather than in memory has some important consequences. Any changes made to a database cause immediate changes to be made to the disk files associated with the database. Therefore, unlike workfiles, once a change is made to a database, there is no possibility of discarding the change and going back to the previously saved version. Because of this, you should take care when modifying a database, and should consider keeping regular backup copies of databases which you modify frequently.

EViews also allows you to deal with a variety of foreign format databases through the same interface provided to EViews' native format databases. Foreign databases can have many different forms, including files on disk, or data made available through some sort of network server. See "Foreign Format Databases" on page 289 for a discussion of the different types of foreign databases that EViews can access.

Creating a Database

To create a database, simply select **File/New/Database...** from the main menu.

For a native EViews database, simply enter a name for the database in the field labeled **DB File name/path**, then click on the button marked **OK**. This will create a new EViews database in the current path.



To create a database in a different directory, you can enter the full path and database name in the **DB File name/path** edit field. Alternatively, you can browse to the desired directory. Simply click on the **Browse Files** button to call

up the common file dialog, and then navigate to the target directory. Enter the name of the new database in the **File name** edit field, then click on the **OK** button to accept the information and close the file dialog. EViews will put the new path and filename in the **DB File name/path** edit field.

The **Database/File Type** field allows you to create different types of databases. See "Foreign Format Databases" on page 289 for a discussion of working with different database types.

The **Open As** field allows you to specify the shorthand that will be associated with this database. A shorthand is a short text label which is used to refer to the database in commands and programs. If you leave this field blank, a default shorthand will be assigned automatically (see "Database Shorthands" on page 265).

The **Browse Registry** and **Add to Registry** buttons provide a convenient way to recall information associated with a previously registered database or to include the new database in the database registry (see "The Database Registry" on page 275).

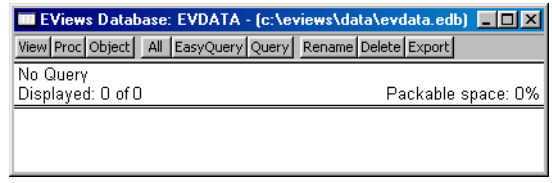
A database can also be created from the command line or in a program using the command:

```
dbcreate db_name
```

where db_name is the name of the database using the same rules given above.

The Database Window

When you create a new database, a database window will open on the screen.



The database window provides a graphical interface which allows you to query the database, copy-and-paste objects to and from your workfile, and perform basic maintenance on the database. Note that some database operations can also be carried out directly without first opening the database window.

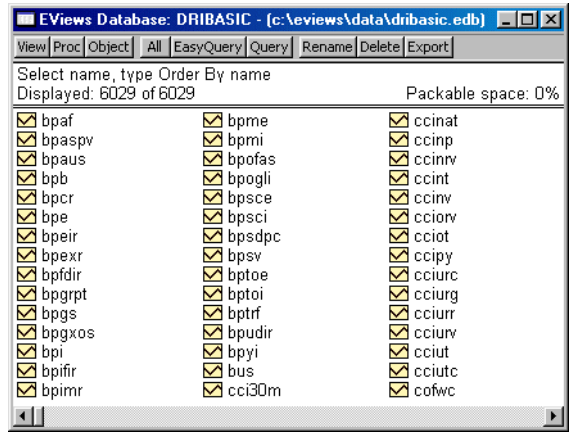
To open a database window for an existing database, select **File/Open/Database…** from the main menu. The same dialog will appear as was used during database creation. To open an EViews database, use the **Browse Files** button to select a file using the common file dialog, then click on **OK** to open the file. A new window should appear representing the open database.

From the command line or in a program, you can open a database window by typing:

```
dbopen db_name
```

Unlike a workfile window, a database window does not display the contents of the database when it is first opened, although it does tell you how many objects are in the database. The second line of the window text shows the number of objects currently displayed (zero when the window is first opened) followed by the total number of objects stored in the database.

You can bring up an alphabetical listing of every object in the database by clicking on the **All** button:



As for a workfile, each object is preceded by a small icon that identifies the type of the object. When performing an **All** query, no other information about the object is visible. However, by double clicking on an object you can bring up a full description of the object including its name, type, modification date, frequency, start and end date (for series), and label.

For large databases, the **All** button generally displays too many objects and not enough information about each object. The database query features ("Querying the Database" on page 277) allow you to control precisely which objects should be displayed, and what information about each object should be visible. The text form of the query currently being displayed is always visible in the top line of the database window.

When working with foreign databases, the object names may appear in color to indicate that they are illegal names or that an alias has been attached to an object name (see "Object Aliases and Illegal Names" on page 285).

The "Packable space" field in the database window displays the percentage of unused space in the database that can be recovered by a database pack operation (see "Packing the Database" on page 288).

A brief technical note: having a database window open in EViews generally does not keep a file open at the operating system level. EViews will normally open files only when it is performing operations on those files. Consequently, multiple users may have a database open at the same time and can perform operations simultaneously. There are some limits imposed by the fact that one user cannot read from a database that another user is writing to at the same time. However, EViews will detect this situation and continue to retry the

operation until the database becomes available. If the database does not become available within a specified time, EViews will generate an error stating that a "sharing violation" on the database has occurred.

For some foreign formats, even minor operations on a database may require full rewriting of the underlying file. In these cases, EViews will hold the file open as long as the database window is open in order to improve efficiency. The formats that currently behave this way are Aremos TSD files, RATS portable files and TSP portable files. When using these formats, only one user at a time may have an open database window for the file.

Database Shorthands

In many situations, EViews allows you to prefix an object name with a database identifier to indicate where the series is located. These database identifiers are referred to as "shorthands". For example, the command:

```
fetch db1::x db2::y
```

indicates to EViews that the object named X is located in the database with the shorthand db1 and the object named y is located in the database with the shorthand db2.

Whenever a database is opened or created, it is assigned a shorthand. The shorthand can be specified by the user in the **Open as** field when opening a database, or using the "As" clause in the `dbopen` command (see `dbopen` (p. 266) in the *Command and Programming Reference*). If a shorthand is explicitly specified when opening a database, an error will occur if the shorthand is already in use.

If no shorthand is provided by the user, a shorthand is assigned automatically. The default value will be the name of the database after any path or extension information has been removed. If this shorthand is already in use, either because a database is already open with the same name, or because an entry in the database registry already uses the name, then a numerical suffix is appended to the shorthand, counting upwards until an unused shorthand is found.

For example, if we open two databases with the same name in a program:

```
dbopen test.edb
dbopen test.dat
```

then the first database will receive the shorthand "TEST" and the second database will receive the shorthand "TEST1". If we then issue the command:

```
fetch test::x
```

the object X will be fetched from the EViews database TEST.EDB. To fetch X from the Haver database TEST.DAT we would use:

```
fetch test1::x
```

To minimize confusion, you should assign explicit shorthands to databases whenever ambiguity could arise. For example, we could explicitly assign the shorthand TEST_HAVER to the second database by replacing the second dbopen command with:

```
dbopen test.dat as test_haver
```

The shorthand attached to a database remains in effect until the database is closed. The shorthand assigned to an open database is displayed in the title bar of the database window.

The Default Database

In order to simplify common operations, EViews uses the concept of a *default database*. The default database is used in several places, the most important of which is as the default source or destination for store or fetch operations when an alternative database is not explicitly specified.

The default database is set by opening a new database window, or by clicking on an already open database window if there are multiple databases open on the screen. The name of the default database is listed in the status line at the bottom of the main EViews window (see Chapter 4, "Object Basics", on page 73, for details). The concept is similar to that of the current workfile with one exception: when there are no currently open databases there is still a default database; when there are no currently open workfiles, the current workfile is listed as "none."

EViews .DB? files

Early versions of EViews and MicroTSP supported a much more limited set of database operations. Objects could be stored on disk in individual files, with one object per file. Essentially, the disk directory system was used as a database and each database entry had its own file. These files had the extension ".DB" for series, and .DB followed by an additional character for other types of objects. EViews refers to these collectively as .DB? files.

While the new database features added to EViews provide a superior method of archiving and managing your data, .DB? files provide backward compatibility and a convenient method of distributing data to other programs. Series .DB files are now supported by a large number of programs including TSP, RATS, and SHAZAM. Additionally, some organizations such as the National Bureau of Economic Research (NBER), distribute data in .DB format.

Working with Objects in Databases

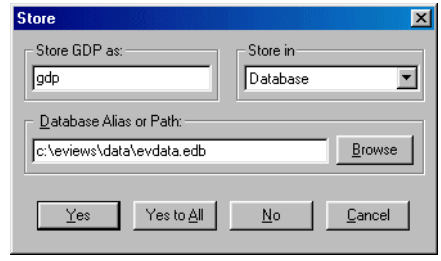
Since databases are simply containers of other EViews objects, most of your work with databases will involve moving objects into and out of them. The sections on storing, fetching and exporting objects discuss different ways of doing this.

You will also need to manage the objects inside a database. You can create duplicate copies of objects, change their names, or remove them from the database entirely. The sections on copying, renaming and deleting discuss how these operations can be carried out.

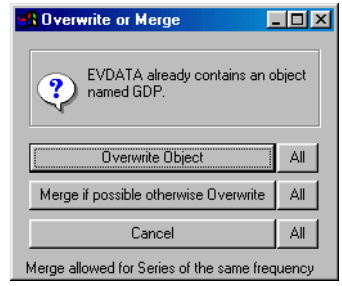
Storing Objects in the Database

An object may be stored in a database in a number of ways. If you have a workfile open on the screen and would like to store objects contained inside it into a database, just select the objects from the workfile window with the mouse, then click on the **Store** button in the workfile toolbar. A sequence of dialogs will come up, one for each object selected, which provide a number of options for renaming the object and determining where the object should be stored.

By default, the object will be stored in the default database with the name used as the workfile. Click **Yes** to store the specified object. If you are storing more than one object, EViews will allow you to select **Yes-to-All** to store all of the objects using the current settings.



If you would like to store the object with a different name, simply type the new name over the old name in the **Store object_name as** edit box. If you would like to store the object in a different database, either enter the name of the new database in the text box marked **Database Alias or Path** (see "The Database Registry" on page 275 for an explanation of database aliases), or click on the button marked **Browse** to select the database name interactively. To store the object to disk as an EViews .DB? file, click on the arrow to the right of the field labeled **Store in** and select **Individual .DB? files**. You may then specify a path in which to place the file using the field labeled **Path for DB files**.

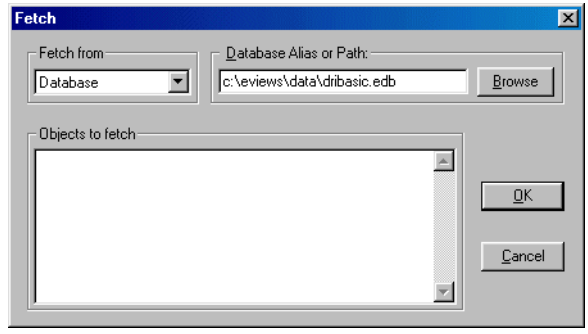


If there is already an existing object in the database with the same name, EViews will display a dialog. The first and last of the three options should be self explanatory. The second option may only be used if the object you are storing from the workfile and the object already in the database are both series of the same frequency. In this case, EViews will merge the data from the two series so that the new series in the database has all the observations from the series being stored, as well as any observations from the existing series which have not been overwritten. For example, if the existing series in the database is an annual series from 1950 to 1990, and the series being stored is an annual series from 1980 to 1995, the new series will run from 1950 to 1995, with data from the existing series for 1950 to 1979, and data from the new series for 1980 to 1995.

Fetching Objects from the Database

There are a number of ways to fetch objects from a database, most of which are similar to the methods for storing.

The first method is to click on the button marked **Fetch** on the toolbar of the workfile into which you would like to fetch the object. A dialog will come up which is similar to the dialog for store:



The dialog allows you to specify the names of the objects to fetch, and the database or directory from which to retrieve them.

Enter the names of the objects you would like to fetch in the field **Objects to Fetch**. Alternatively, you can use the mouse to select objects from the workfile window before clicking on the **Fetch** button, in which case the names of these objects will appear automatically. The fields labeled **Database Alias or Path** and **Fetch from** are the same as for the store dialog with one exception. In addition to **EViews Database** and **Individual .DB? files, Fetch from** has an option titled **Search Databases.** This option tells EViews to search multiple databases for objects which match the specified names. To use this option, you must first define a search order in the database registry (see "The Database Registry" on page 275).

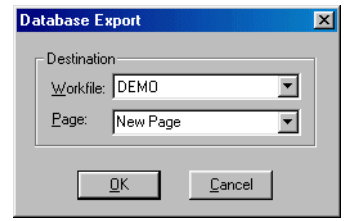
When you click on **OK**, EViews will fetch all the objects. If an object which is being fetched is already contained in the workfile, a dialog will appear asking whether to replace the object or not. Click on **Yes** to replace the object in the workfile or **No** to leave the object in the workfile unchanged.

Because a workfile has a fixed frequency and range, fetching a series into a workfile may cause the data in the series to be modified to match the frequency and range of the workfile (see "Frequency Conversion" on page 115). Be aware that loading a series into a workfile then saving it back into the database can cause truncation and frequency conversion of the series stored in the database.

Object/Update selected from DB... from the workfile toolbar is the same as **Fetch** except that there is no overwrite warning message. If the object in the database is the same type as the one in the workfile, it is automatically overwritten. If it is of a different type, the fetch does not proceed. Update is also available from the **Object** button in individual object windows.

Database Export

You can also move data into a workfile from the database window. From an open database window, select the objects you would like to copy using the mouse, then click on the button marked **Export** in the toolbar at the top of the database window. The **Database Export** dialog will appear on the screen:



When you click on the down arrow on the right of the field labeled **Workfile**, a list of all workfiles that are currently open will appear from which you may choose the workfile into which you would like to copy the objects. In addition, you may use the **Page** drop down menu to select an existing page in the selected workfile, or to create a new page. Clicking on the button marked **OK** will copy the selected objects to specified page of the selected workfile.

There is an extra option in the list of open workfiles for specifying a new workfile as your copy destination. If you select **New Workfile**, EViews will create a new workfile containing the objects you have selected. After you click on **OK**, a second dialog will appear in which you can set the frequency and range of the workfile to be created. The default frequency is set to the lowest frequency of any of the objects selected, and the default range is set to cover all the data points contained in the objects. Clicking on **OK** will open a new workfile window and copy the selected objects into it, performing frequency conversion where necessary.

Copying Objects

In addition to the above methods for moving objects, EViews provides general support for the copying of objects between any two EViews container objects (workfiles or databases). You may use these features to move objects between two databases or between two workfiles, to create duplicate copies of objects within a workfile or database, or as an alternative method for store and fetch.

Copy-and-Paste

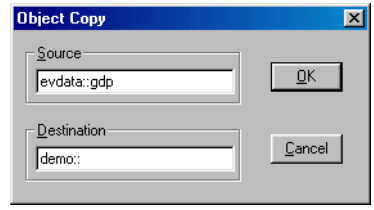
For copying objects between containers, the procedure is very similar no matter what types of container objects are involved. Before you start, make sure that the windows for both containers are open on the screen. In the container *from which* you would like to copy the objects, select the objects then click on **Edit/Copy** in the EViews program menu. Click on the container object *into which* you would like to paste the objects, then select **Edit/Paste** or **Edit/Paste Special...** from the EViews program menu.

Depending on the types of the two containers, you may be presented with one or more dialogs. If, for example, you are performing a copy to or from a database, and click on **Edit/Paste**, the standard **Store** or **Fetch** dialogs will appear as if you had carried out the operations using the toolbar buttons on the workfile window. If you click on **Edit/Paste Special...**, an alternate dialog will be displayed, allowing you to override the default frequency conversion methods.

If, instead, you are copying between two workfiles, selecting **Edit/Paste** will simply copy the series using the default frequency conversion if necessary. You will only be prompted with a dialog if there is name collision. Selecting **Edit/Paste Special...** will display a dialog allowing you to override the default conversion methods.

Copy Procedure

You may perform similar operations using the object copy procedure. From the main menu select **Object/Copy** (this may appear as **Object/Copy selected...**). The **Object Copy** dialog will be displayed.



The **Source** field specifies the object or objects you would like to copy, the **Destination** field specifies where you would like to copy them and what names they should be given.

The **Source** field should be filled in with an expression of the form:

```
source_db::source_pattern
```

where `source_db::` is optional, and indicates which database the objects should be copied from (if no database name is supplied, the source is taken to be the default workfile), and `source_pattern` is either a simple object name or a name pattern. A name pattern may include the wildcard characters "?" which matches any single character, and "*" which matches zero or more characters.

The **Destination** field should be filled in with an expression of the form:

```
dest_db::dest_name
```

where `dest_db::` is again optional, and indicates which database the objects should be copied to (if no database name is supplied, the destination is taken to be the default workfile), and `dest_name`, which is also optional, is the name to be given to the new copy of the object. If no name is given, the object will be copied with its existing name. If a pattern was used when specifying the source, a pattern must also be used when specifying the destination (see "Source and Destination Patterns" on page 946).

For example, to copy an object from the database DB1 to the database DB2, keeping the existing name, you would fill in the dialog:

source: db1::object_name

destination: db2::

where OBJECT_NAME is the original name as displayed by EViews.

To copy all the objects in the database DB1 beginning with the letter X into the current workfile, changing the names so that they begin with Y, you would fill in the dialog

source: db1::x*

destination: y*

To make a duplicate copy of the object named ABC in the database DB1, giving it the new name XYZ, you would fill in the dialog:

source: db1::abc

destination: db1::xyz

Renaming Objects in the Database

You may rename an object in the database by selecting the object in an open database window, then clicking on the button marked **Rename** in the database window toolbar. A dialog will come up in which you can modify the existing name or type in a new name. You can rename several objects at the same time using wildcard patterns and the `rename` command.

Deleting Objects From the Database

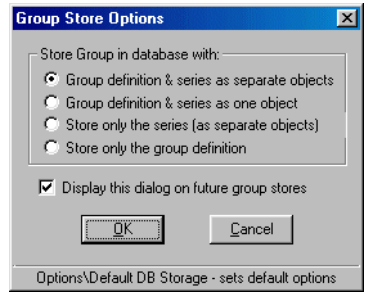
To delete objects from the database, select the objects in an open database window, then click on the button marked **Delete** on the database window toolbar. You may delete several objects at the same time using wildcard patterns. There is also a `delete` command. See `delete` (p. 272) in the *Command and Programming Reference* for details.

Store, Fetch, and Copy of Group Objects

A group object in EViews is essentially a list of series names that form the group. The data of each series are contained in the series object, not in the group object. When you do a store, fetch, or copy operation on a group object, an issue arises as to whether you want to do the operation on each of the series or to the group definition list.

Storing a Group Object

When you store a group object to a database, there are four available options:


Group Store Options
Store Group in database with:
Group definition & series as separate objects
Group definition & series as one object
Store only the series (as separate objects)
Store only the group definition
Display this dialog on future group stores
OK
Cancel
Options\Default DB Storage - sets default options

- Store the group definition and the series as separate objects: stores the group object (only its definition information) and each of its series as separate objects in the database. If any of the series already exist in the database, EViews will ask whether or not to overwrite the existing series if in interactive mode, and will error if in batch mode.

- Store the group definition and the series as one object: stores each series within the group object. A group object that contains series data will have an icon G+ in the database directory window. A group object with only its definition information will have the usual icon G. If you use this option, you can store two different series with the same name (with one of the series as member of a group).

- Store only the series (as separate objects): only stores each series as separate objects in the database. If you want to store a long list of series into a database, you can create a temporary group object that contains those series and issue the store command only once.

- Store only the group definition: stores only the group definition information; none of the series data are stored in the database. This option is useful if you want to update the member data from the database but want to keep the group information (e.g. the dated data table settings) in the group.

By default, EViews will display a dialog asking you to select a group store option every time you store a group object. You can, however, instruct EViews to suppress the dialog

and use the global option setting. Simply click on **Options/Database Storage Defaults...** in the main EViews menu to bring up a dialog that allows you both to set the global storage options, and to suppress the group store option dialog.

Fetching a Group Object

When you fetch a group object to a database, there are three options available:

Group Fetch Options
Group fetch behavior
Fetch both group definition & series
Fetch only the series in the group
Fetch only the group definition
Display this dialog on future group fetches
OK
Cancel
Options\Default DB Storage - sets default options

- Fetch both group definition and the actual series: fetches both group definition and its series as separate objects. If any of the series defined in the group is not found in the database, the corresponding series will be created in the workfile filled with NAs. If any of the series already exist in the workfile, EViews will ask whether or not to overwrite the existing series if in interactive mode, and will error if in batch mode.
- Fetch only the series in the group: only fetches each series defined in the group. If the series exists both within the group object (with a G+ icon) and as a separate series object in the database, the series within the group object will be fetched.
- Fetch only the group definition: fetches only the group definition (but not the series data). If any of the series defined in the group does not exist in the workfile, EViews will create the corresponding series filled with NAs.

You can click on **Options/Database Default Storage Options...** in the main menu to bring up a dialog that allows you both to set the global fetch options, and to suppress the fetch option dialog.

Copying Group Objects between Workfiles and Databases

You can also copy groups between different containers. The options that are available will differ depending on the type of source and destination container:

- Copy from workfile to database: same options as the store operation.
- Copy from database to workfile: same options as the fetch operation.
- Copy from workfile to workfile: both the group definition and series will be copied.
- Copy from database to database. If the group object contains only the group definition (with a G icon), only the group definition will be copied. If the group object also contains its series data (with a G+ icon), then the group will be copied containing the series data and the copied group will also appear with a G+ icon.

Database Auto-Series

We have described how to fetch series into a workfile. There is an alternative way of working with databases which allows you to make direct use of the series contained in a database without first copying the series. The advantage of this approach is that you need not go through the process of importing the data every time the database is revised. This approach follows the model of auto-series in EViews as described in "Auto-series" beginning on page 141.

There are many places in EViews where you can use a series expression, such as log(X), instead of a simple series name, and EViews will automatically create a temporary auto-series for use in the procedure. This functionality has been extended so that you can now directly refer to a series in a database using the syntax:

```
db_name::object_name
```

where `db_name` is the shorthand associated with the database. If you omit the database name and simply prefix the object name with a double colon like this:

```
::object_name
```

EViews will look for the object in the default database.

A simple example is to generate a new series:

```
series lgdp = log(macro_db::gdp)
```

EViews will fetch the series named GDP from the database with the shorthand MACRO_DB, and put the log of GDP in a new series named LGDP in the workfile. It then deletes the series GDP from memory, unless it is in use by another object. Note that the generated series LGDP only contains data for observations within the current workfile sample.

You can also use auto-series in a regression. For example:

```
equation eq1.ls log(db1::y) c log(db2::x)
```

This will fetch the series named Y and X from the databases named DB1 and DB2, perform any necessary frequency conversions and end point truncation so that they are suitable for use in the current workfile, take the log of each of the series, then run the requested regression. Y and X are then deleted from memory unless they are otherwise in use.

The auto-series feature can be further extended to include automatic searching of databases according to rules set in the database registry (see "The Database Registry" on page 275). Using the database registry you can specify a list of databases to search whenever a series you request cannot be found in the workfile. With this feature enabled, the `series` command:

```
series lgdp = log(gdp)
```

looks in the workfile for a series named GDP. If it is not found, EViews will search through the list of databases one by one until a series called GDP is found. When found, the series will be fetched into EViews so that the expression can be evaluated. Similarly, the regression:

```
equation logyeq.ls log(y) c log(x)
```

will fetch Y and X from the list of databases in the registry if they are not found in the workfile. Note that the regression output will label all variables with the database name from which they were imported.

In general, using auto-series directly from the database has the advantage that the data will be completely up to date. If the series in the database are revised, you do not need to repeat the step of importing the data into the workfile. You can simply reestimate the equation or model, and EViews will automatically retrieve new copies of any data which are required.

There is one complication to this discussion which results from the rules which regulate the updating and deletion of auto-series in general. If there is an existing copy of an auto-series already in use in EViews, a second use of the same expression will not cause the expression to be reevaluated (in this case reloaded from the database); it will simply make use of the existing copy. If the data in the database have changed since the last time the auto-series was loaded, the new expression will use the old data.

One implication of this behavior is that a copy of a series from a database can persist for any length of time if it is stored as a member in a group. For example, if you type:

```
show db1::y db2::x
```

this will create an untitled group in the workfile containing the expressions `db1::y` and `db2::x`. If the group window is left open and the data in the database are modified (for example by a `store` or a `copy` command), the group and its window will not update automatically. Furthermore, if the regression:

```
equation logyeq.ls log(db1::y) c log(db2::x)
```

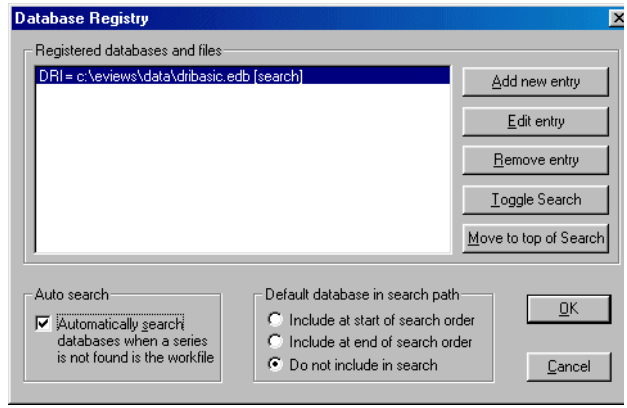
is run again, this will use the copies of the series contained in the untitled group; it will not refetch the series from the database.

The Database Registry

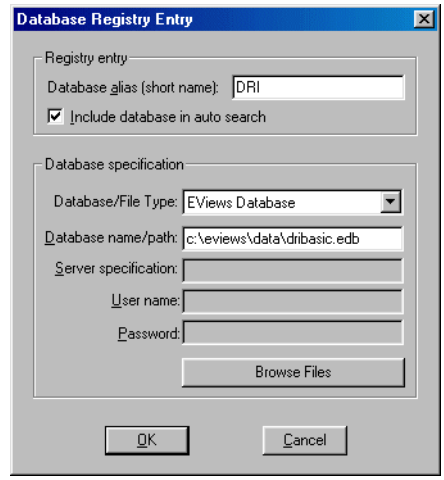
The database registry is a file on disk that manages a variety of options which control database operations. It gives you the ability to assign short alias names that can be used in

place of complete database paths, and also allows you to configure the automatic searching features of EViews.

Options/Database Registry… from the main menu brings up the **Database Registry** dialog allowing you to view and edit the database registry:



The box labeled **Registry Entries** lists the databases that have been registered with EViews. The first time you bring up the dialog, the box will usually be empty. If you click on the **Add new entry** button, a **Database Registry Entry** dialog appears.



There are three things you must specify in the dialog: the full name (including path) of the database, the alias which you would like to associate with the database, and the option for whether you wish to include the database in automatic searches.

The full name and path of the database should be entered in the top edit field. Alternatively, click the **Browse** button to select your database interactively.

The next piece of information you must provide is a *database alias*: a short name that you can use in place of the full database path in EViews commands. The database alias will also be used by EViews to label database auto-series. For example, suppose you have a database named DRIBASIC located in the subdirectory C:\EVIEWS\DATA. The following regression command is legal but awkward:

```
equation eq1.ls c:\eviews\data\dribasic::gdp c
   c:\eviews\data\dribasic::gdp(-1)
```

Long database names such as these also cause output labels to truncate, making it difficult to see which series were used in a procedure.

By assigning full database path and name the alias DRI, we may employ the more readable command:

```
equation eq1.ls dri::gdp c dri::gdp(-1)
```

and the regression output will be labeled with the shorter names. To minimize the possibility of truncation, we recommend the use of short alias names if you intend to make use of database auto-series.

Finally, you should tell EViews if you want to include the database in automatic database searches by checking the **Include in auto search** checkbox. Click on **OK** to add your entry to the list

Any registry entry may be edited, deleted, switched on or off for searching, or moved to the top of the search order by highlighting the entry in the list and clicking the appropriate button to the right of the list box.

The remainder of the **Database Registry** dialog allows you to set options for automatic database searching. The **Auto-search** checkbox is used to control EViews behavior when you enter a command involving a series name which cannot be found in the current workfile. If this checkbox is selected, EViews will automatically search all databases that are registered for searching, before returning an error. If a series with the unrecognized name is found in any of the databases, EViews will create a database auto-series and continue with the procedure.

The last section of the dialog, **Default Database in Search Order**, lets you specify how the default database is treated in automatic database searches. Normally, when performing an automatic search, EViews will search through the databases contained in the **Registry Entries** window in the order that they are listed (provided that the **Include in auto search** box for that entry has been checked). These options allow you to assign a special role to the default database when performing a search.

- **Include at start of search order—**means that the current default database will be searched first, before searching the listed databases.
- **Include at end of search order—**means that the current default database will be searched last, after searching the listed databases.
- **Do not include in search—**means that the current default database will not be searched unless it is already one of the listed databases.

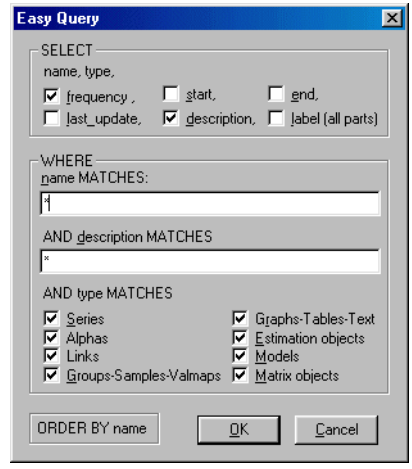
Querying the Database

A great deal of the power of the database comes from its extensive query capabilities. These capabilities make it easy to locate a particular object, and to perform operations on a set of objects which share similar properties.

The query capabilities of the database can only be used interactively from the database window. There are two ways of performing a query on the database: the easy mode and the advanced mode. Both methods are really just different ways of building up a text query to the database. The easy mode provides a simpler interface for performing the most common types of queries. The advanced mode offers more flexibility at the cost of increased complexity.

Easy Queries

To perform an easy query, first open the database, then click on the **EasyQuery** button in the toolbar at the top of the database window. The Easy Query dialog will appear containing two text fields and a number of check boxes:



There are two main sections to this dialog: **Select** and **Where**. The **Select** section determines which fields to display for each object that meets the query condition. The **Where** section allows you to specify conditions that must be met for an object to be returned from the query. An Easy Query allows you to set conditions on the object name, object description, and/or object type.

The two edit fields (**name** and **description**) and the set of check boxes (object **type**) in the **Where** section provide three filters of objects that are returned from the query to the database. The filters are applied in sequence (using a logical 'and' operation) so that objects in the database must meet all of the criteria selected in order to appear in the results window of the query.

The **name** and **description** fields are each used to specify a *pattern expression* that the object must meet in order to satisfy the query. The simplest possible pattern expression consists of a single pattern. A pattern can either be a simple word consisting of alphanumeric characters, or a pattern made up of a combination of alphanumeric characters and the wildcard symbols "?" and "*", where "?" means to match any one character and "*" means to match zero or more characters. For example:

```
pr?d*ction
```

would successfully match the words production, prediction, and predilection. Frequently used patterns include "s*" for words beginning in "S", "*s" for words ending in "S", and "*s*" for words containing "S". Upper or lower case is not significant when searching for matches.

Matching is done on a word-by-word basis, where at least one word in the text must match the pattern for it to match overall. Since object names in a database consist of only a single word, pattern matching for names consists of simply matching this word.

For descriptions, words are constructed as follows: each word consists of a set of consecutive alphanumeric characters, underlines, dollar signs, or apostrophes. However, the following list words are explicitly ignored: "a", "an", "and", "any", "are", "as", "be", "between", "by", "for", "from", "if", "in", "is", "it", "not", "must", "of", "on", "or", "should", "that", "the", "then", "this", "to", "with", "when", "where", "while". (This is done for reasons of efficiency, and to minimize false matches to patterns from uninteresting words). The three words "and", "or", and "not" are used for logical expressions.

For example:

```
bal. of p'ment: seas.adj. by X11
```

is broken into the following words: "bal", "p'ment", "seas", "adj", and "x11". The words "of" and "by" are ignored.

A pattern expression can also consist of one or more patterns joined together with the logical operators "and", "or" and "not" in a manner similar to that used in evaluating logical expressions in EViews. That is, the keyword `and` requires that both the surrounding conditions be met, the keyword `or` requires that either of the surrounding conditions be met, and the keyword `not` requires that the condition to the right of the operator is not met. For example:

```
s* and not *s
```

matches all objects which contain words which begin with, but do not end with, the letter "S".

More than one operator can be used in an expression, in which case parentheses can be added to determine precedence (the order in which the operators are evaluated). Operators inside parentheses are always evaluated logically prior to operators outside parentheses. Nesting of parentheses is allowed. If there are no parentheses, the precedence of the operators is determined by the following rules: `not` is always applied first; `and` is applied second; and `or` is applied last. For example:

```
p* or s* and not *s
```

matches all objects which contain words beginning with P, or all objects which contain words which begin with, but do not end with, the letter S.

The third filter provided in the **Easy Query** dialog is the ability to filter by object type. Simply select the object types which you would like displayed, using the set of check boxes near the bottom of the dialog.

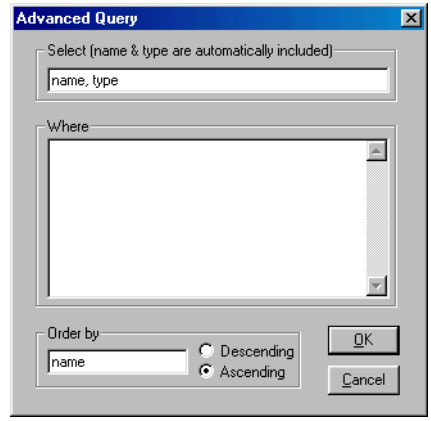
Advanced Queries

Advanced queries allow considerably more control over both the filtering and the results which are displayed from a query. Because of this flexibility, advanced queries require some understanding of the structure of an EViews database to be used effectively.

Each object in an EViews database is described by a set of fields. Each field is identified by a name. The current list of fields includes:

name	The name of the object.
type	The type of the object.
last_write	The time this object was last written to the database.
last_update	The time this object was last modified by EViews.
freq	The frequency of the data contained in the object.
start	The date of the first observation contained in the object.
end	The date of the last observation contained in the object.
obs	The number of data points stored in the series (including missing values).
description	A brief description of the object.
source	The source of the object.
units	The units of the object.
remarks	Additional remarks associated with the object.
history	Recent modifications of the object by EViews.
display_name	The EViews display name.

An advanced query allows you to examine the contents of any of these fields, and to select objects from the database by placing conditions on these fields. An advanced query can be performed by opening the database window, then clicking on the button marked **Query** in the toolbar at the top of the window. The Advanced Query dialog is displayed.



The first edit field labeled **Select:** is used to specify a list of all the fields that you would like displayed in the query results. Input into this text box consists of a series of field names separated by commas. Note that the name and type fields are always fetched automatically.

The ordering of display of the results of a query is determined by the **Order By** edit field. Any field name can be entered into this box, though some fields are likely to be more useful than others. The description field, for example, does not provide a useful ordering of the objects. The **Order By** field can be useful for grouping together objects with the same value of a particular field. For example, ordering by `type` is an effective way to group together the results so that objects of the same type are placed together in the database window. The **Ascending** and **Descending** buttons can be used to reverse the ordering of the objects. For example, to see objects listed from those most recently written in the database to those least recently written, one could simply sort by the field `last_write` in **Descending** order.

The **Where** edit field is the most complicated part of the query. Input consists of a logical expression built up from conditions on the fields of the database. The simplest expression is an operator applied to a single field of the database. For example, to search for all series which are of monthly or higher frequencies (where higher frequency means containing more observations per time interval), the appropriate expression is:

```
freq >= monthly
```

Field expressions can also be combined with the logical operators `and`, `or` and `not` with precedence following the same rules as those described above in the section on easy queries. For example, to query for all series of monthly or higher frequencies which begin before 1950, we could enter the expression:

```
freq >= monthly and start < 1950
```

Each field has its own rules as to the operators and constants which can be used with the field.

Name

The name field supports the operators " < ", " < = ", " > ", " > = ", " = ", and " < > " to perform typical comparisons on the name string using alphabetical ordering. For example,

```
name >= c and name < m
```

will match all objects with names beginning with letters from C to L. The name field also supports the operator "matches". This is the operator which is used for filtering the name field in the easy query and is documented extensively in the previous section. Note that if `matches` is used with an expression involving more than one word, the expression must be contained in quotation marks. For example,

```
name matches "x* or y*" and freq = quarterly
```

is a valid query, while

```
name matches x* or y* and freq = quarterly
```

is a syntax error because the part of the expression that is related to the `matches` operator is ambiguous.

Type

The type field can be compared to the following object types in EViews using the " = " operator: `sample`, `equation`, `graph`, `table`, `text`, `program`, `model`, `system`, `var`, `pool`, `sspace`, `matrix`, `group`, `sym`, `matrix`, `vector`, `coef`, `series`. Relational operators are defined for the type field, although there is no particular logic to the ordering. The ordering can be used, however, to group together objects of similar types in the **Order By** field.

Freq

The frequency field has one of the following values:

u	Undated
a	Annual
s	Semiannual
q	Quarterly
m	Monthly
w	Weekly
5	5 day daily
7	7 day daily

Any word beginning with the letter above is taken to denote that particular frequency, so that monthly can either be written as "m" or "monthly". Ordering over frequencies is defined so that a frequency with more observations per time interval is considered "greater" than a series with fewer observations per time interval. The operators " < ", " > ", " < = ", " > = ", " = ", " < > " are all defined according to these rules. For example,

```
freq <= quarterly
```

will match objects whose frequencies are quarterly, semiannual, annual or undated.

Start and End

Start and end dates use the following representation. A date from an annual series is written as an unadorned year number such as "1980". A date from a semiannual series is written as a year number followed by an "S" followed by the six month period, for example "1980S2". The same pattern is followed for quarterly and monthly data using the letters "Q" and "M" between the year and period number. Weekly, 5-day daily, and 7-day daily data are denoted by a date in the format:

```
mm/dd/yyyy
```

where `m` denotes a month digit, `d` denotes a day digit, and `y` denotes a year digit.

Operators on dates are defined in accordance with calendar ordering where an earlier date is less than a later date. Where a number of days are contained in a period, such as for monthly or quarterly data, an observation is ordered according to the first day of the period. For example:

```
start <= 1950
```

will include dates whose attributed day is the first of January 1950, but will not include dates which are associated with other days in 1950, such as the second, third, or fourth quarter of 1950. However, the expression:

```
start < 1951
```

would include all intermediate quarters of 1950.

Last_write and Last_update

As stated above, `last_write` refers to the time the object was written to disk, while `last_update` refers to the time the object was last modified inside EViews. For example, if a new series was generated in a workfile, then stored in a database at some later time, `last_write` would contain the time that the store command was executed, while `last_update` would contain the time the new series was generated. Both of these fields contain date and time information which is displayed in the format:

```
mm/dd/yyyy hh:mm
```

where `m` represents a month digit, `d` represents a day digit, `y` represents a year digit, `h` represents an hour digit and `m` represents a minute digit.

The comparison operators are defined on the time fields so that earlier dates and times are considered less than later dates and times. A typical comparison has the form:

```
last_write >= mm/dd/yyyy
```

A day constant always refers to twelve o'clock midnight at the beginning of that day. There is no way to specify a particular time during the day.

Description, Source, Units, Remarks, History, Display_name

These fields contain the label information associated with each object (which can be edited using the Label view of the object in the workfile). Only one operator is available on these fields, the `matches` operator, which behaves exactly the same as the description field in the section on easy queries.

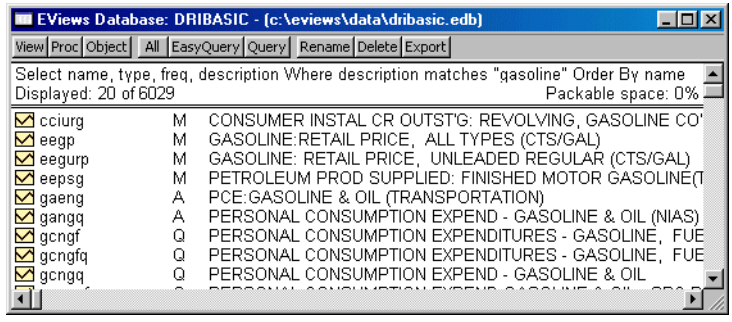
Query Examples

Suppose you are looking for data related to gasoline consumption and gasoline prices in the database named DRIBASIC. First open the database: click **File/Open**, select **Files of type: Database .edb** and locate the database. From the database window, click **Query** and fill in the **Advanced Query** dialog as follows:

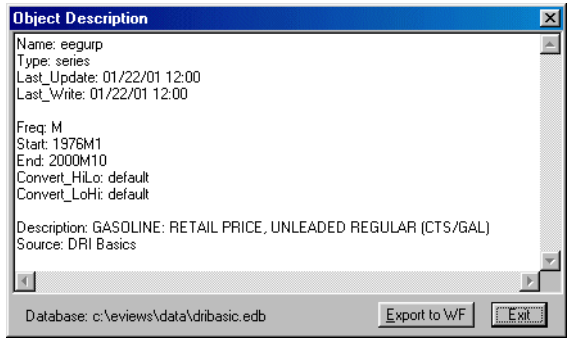
Select: `name, type, freq, description`

Where: `description matches gasoline`

If there are any matches, the results are displayed in the database window similar to the following:


EViews Database: DRIBASIC - (c:\eviews\data\dribasic.edb)
View Proc Object All EasyQuery Query Rename Delete Export
Select name, type, freq, description Where description matches "gasoline" Order By name
Displayed: 20 of 6029
Packable space: 0%
cciurg M CONSUMER INSTAL CR OUTST'G: REVOLVING, GASOLINE CO
eegp M GASOLINE:RETAIL PRICE, ALL TYPES (CTS/GAL)
eegurp M GASOLINE: RETAIL PRICE, UNLEADED REGULAR (CTS/GAL)
eepsg M PETROLEUM PROD SUPPLIED: FINISHED MOTOR GASOLINE(T
gaeng A PCE:GASOLINE & OIL (TRANSPORTATION)
gangq A PERSONAL CONSUMPTION EXPEND - GASOLINE & OIL (NIAS)
gcngf Q PERSONAL CONSUMPTION EXPENDITURES - GASOLINE, FUE
gcngfq Q PERSONAL CONSUMPTION EXPENDITURES - GASOLINE, FUE
gcngq Q PERSONAL CONSUMPTION EXPEND - GASOLINE & OIL

To view the contents of all fields of an item, double click on its name. EViews will open an **Object Description** window that looks as follows:



To further restrict your search to series with at least quarterly frequency and to display the start and end dates of the results, click **Query** and again and modify the fields as follows:

Select: `name, type, start, end, description`

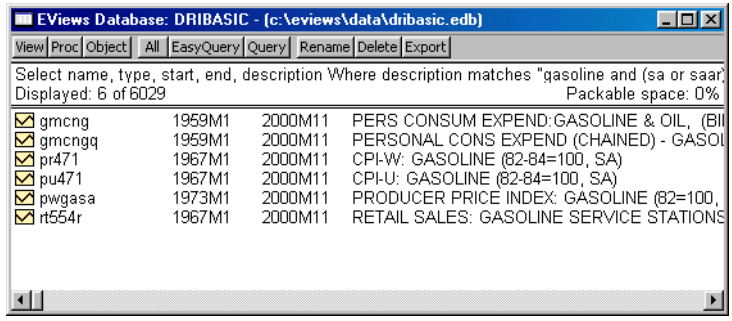
Where: `description matches gasoline and freq>=q`

If you are interested in seasonally adjusted series, which happen to contain `sa` or `saar` in their description in this database, further modify the fields to

Select: `name, type, start, end, description`

Where: `description matches "gasoline and (sa or saar)" and freq>=q`

The display of the query results now looks as follows:


EViews Database: DRIBASIC - (c:\eviews\data\dribasic.edb)
View Proc Object All EasyQuery Query Rename Delete Export
Select name, type, start, end, description Where description matches "gasoline and (sa or saar)
Displayed: 6 of 6029
Packable space: 0%
gmcng 1959M1 2000M11 PERS CONSUM EXPEND:GASOLINE & OIL, (BI
gmcngq 1959M1 2000M11 PERSONAL CONS EXPEND (CHAINED) - GASOL
pr471 1967M1 2000M11 CPI-W: GASOLINE (82-84=100, SA)
pu471 1967M1 2000M11 CPI-U: GASOLINE (82-84=100, SA)
pwgasa 1973M1 2000M11 PRODUCER PRICE INDEX: GASOLINE (82=100,
rt554r 1967M1 2000M11 RETAIL SALES: GASOLINE SERVICE STATIONS

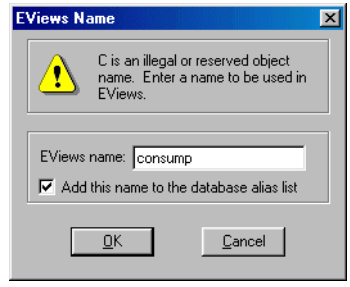
Object Aliases and Illegal Names

When working with a database, EViews allows you to create a list of aliases for each object in the database so that you may refer to each object by a different name. The most important use of this is when working with a database in a foreign format where some of the names used in the database are not legal EViews object names. However, the aliasing fea-

tures of EViews can also be used in other contexts, such as to assign a shorter name to a series with an inconveniently long name.

The basic idea is as follows: each database can have one or more *object aliases* associated with it where each alias entry consists of the name of the object in the database and the name by which you would like it to be known in EViews.

The easiest way to create an object alias for an illegal name is to attempt to fetch the object with the illegal name into EViews. If you are working with query results, you can tell which object names are illegal because they will be displayed in the database window in red. When you try to fetch an object with an illegal name, a dialog will appear.

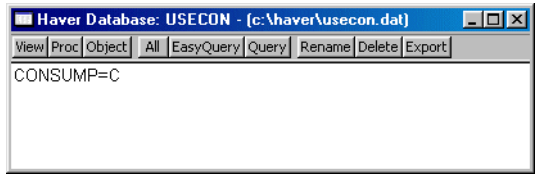

EViews Name
C is an illegal or reserved object name. Enter a name to be used in EViews.
EViews name: consump
Add this name to the database alias list
OK
Cancel

The field labeled **EViews Name** initially contains the illegal name of the database object. You should edit this to form a legal EViews object name. In this example, we could change the name C to CONSUMP. The checkbox labeled **Add this name to the database alias list** (which is not checked by default), determines whether you want to create a permanent association between the name you have just typed and the illegal name. If you check the box, then whenever you use the edited object name in the future, EViews will take it to refer to the underlying illegal name. The edited name acts as an *alias* for the underlying name. It is as though you had renamed the object in the database to the new legal name, except that you have not actually modified the database itself, and your changes will not affect other users of the database.

When EViews displays an object in the database window for which an alias has been set, EViews will show the alias, rather than the underlying name of the object. In order to indicate that this substitution has been done, EViews displays the name of the aliased object in blue.

Creating an alias can cause *shadowing* of object names. Shadowing occurs when you create an alias for an object in the database, but the name you use as an alias is the name of another object in the database. Because the existence of the alias will stop you from accessing the other object, that object is said to be shadowed. To indicate that an object name being displayed has been shadowed, EViews displays the name of shadowed objects in green. You will not be able to fetch an object which has been shadowed without modifying either its name or the alias which is causing it to be shadowed. Even if the shadowed series is explicitly selected with the mouse, operations performed on the series will use the series with the conflicting alias, not the shadowed series.

You can view a list of the aliases currently defined for any database by clicking on the **View** button at the top of the database window, then selecting **Object Aliases**. A list of all the aliases will be displayed in the window.



Each line represents one alias attached to the database and follows the format:

```
alias = database_object_name
```

You can edit the list of aliases to delete unwanted entries, or you can type in, or cut-and-paste, new entries into the file. You must follow the rule that both the set of aliases and the set of database names do not contain any repeated entries. (If you do not follow this rule, EViews will refuse to save your changes). To save any modifications you have made, simply switch back to the **Object Display** view of the database. EViews will prompt you for whether you want to save or discard your edits.

The list of currently defined database aliases for all databases is kept in the file OBALIAS.INI in the EViews installation directory. If you would like to replicate a particular set of aliases onto a different machine, you should copy this file to the other machine, or use a text editor to combine a portion of this file with the file already in use on the other machine. You must exit and restart EViews to be sure that EViews will reread the aliases from the file.

Maintaining the Database

In many cases an EViews database should function adequately without any explicit maintenance. Where maintenance is necessary, EViews provides a number of procedures to help you perform common tasks.

Database File Operations

Because EViews databases are spread across multiple files, all of which have the same name but different extensions, simple file operations like copy, rename and delete require multiple actions if performed outside of EViews. The **Proc** button in the database window toolbar contains the procedures **Copy the database**, **Rename the database**, and **Delete the database** that carry out the chosen operation on all of the files that make up the database.

Note that file operations do not automatically update the database registry. If you delete or rename a database that is registered, you should either create a new database with the same name and location, or edit the registry.

Packing the Database

If many objects are deleted from an EViews database without new objects being inserted, a large amount of unused space will be left in the database. In addition, if objects are frequently overwritten in the database, there will be a tendency for the database to grow gradually in size. The extent of growth will depend on the circumstances, but a typical database is likely to stabilize at a size around 60% larger than what it would be if it were written in a single pass.

A database can be compacted down to its minimum size by using the pack procedure. Simply click on the button marked **Proc** in the toolbar at the top of the database window, then select the menu item **Pack the Database**. Depending on the size of the database and the speed of the computer which you are using, performing this operation may take a significant amount of time.

You can get some idea of the amount of space that will be reclaimed during a pack by looking at the Packable Space percentage displayed in the top right corner of the database window. A figure of 30%, for example, indicates that roughly a third of the database file consists of unused space. A more precise figure can be obtained from the **Database Statistics** view of a database. The number following the label "unused space" gives the number of unused bytes contained in the main database file.

Dealing with Errors

EViews databases are quite robust, so you should not experience problems working with them on a regular basis. However, as with all computer files, hardware or operating system problems may produce conditions under which your database is damaged.

The best way to protect against damage to a database is to make regular backup copies of the database. This can be performed easily using the **Copy the Database** procedure documented above. EViews provides a number of other features to help you deal with damaged databases.

Damaged databases can be divided into two basic categories depending on how severely the database has been damaged. A database which can still be opened in a database window but generates an error when performing some operations may not be severely damaged and may be reparable. A database which can no longer be opened in a database window is severely damaged and will need to be rebuilt as a new database.

EViews has two procedures designed for working with databases which can be opened: **Test Database Integrity** and **Repair Database**. Both procedures are accessed by clicking

on the button marked **Proc** in the database window toolbar, then selecting the appropriate menu item.

Test Database Integrity conducts a series of validity checks on the main database and index files. If an error is detected, a message box will be displayed, providing some information as to the type of error found and a suggestion as to how it might be dealt with. Because testing performs a large number of consistency checks on the database files, it may take considerable time to complete. You can monitor its progress by watching the messages displayed in the status line at the bottom of the EViews window. Testing a database does not modify the database in any way, and will never create additional damage to a database.

Repair Database will attempt to automatically detect and correct simple problems in the database. Although care has been taken to make this command as safe as possible, it will attempt to modify a damaged database, so it is probably best to make a back up copy of a damaged database before running this procedure.

Rebuilding the Database

If the database is badly corrupted, it may not be possible for it to be repaired. In this case, EViews gives you the option of building a new database from the old one using the `dbrebuild` command. This operation can only be performed from the command line (since it may be impossible to open the database). The command is:

```
dbrebuild old_dbname new_dbname
```

The `dbrebuild` command does a low level scan through the main data file of the database `old_dbname` looking for any objects which can be recovered. Any such objects are copied into the new database `new_dbname`. This is a very time consuming process, but it will recover as much data as possible from even heavily damaged files.

Foreign Format Databases

While most of your work with databases will probably involve using EViews native format databases, EViews also gives you the ability to access data stored in a variety of other formats using the same database interface. You can perform queries, copy objects to and from workfiles and other databases, rename and delete objects within the database, add databases to your search path, and use EViews' name aliasing features, all without worrying about how the data are stored.

When copying objects, EViews preserves not only the data itself, but as much as possible of any date information and documentation associated with the object. Missing values are translated automatically.

To Convert Or Not To Convert?

Although EViews allows you to work with foreign files in their native format, in some cases you may be better off translating the entire foreign file into EViews format. If necessary, you can then translate the entire file back again when your work is complete. EViews native databases have been designed to support a certain set of operations efficiently, and while access to foreign formats has been kept as fast as possible, in some cases there will be substantial differences in performance depending on the format in use.

One significant difference is the time taken to search for objects using keywords in the description field. If the data is are EViews format, EViews can typically query databases containing tens of thousands of series in a couple of seconds. When working with other formats, you may find that this same operation takes much longer, with the time increasing substantially as the database grows.

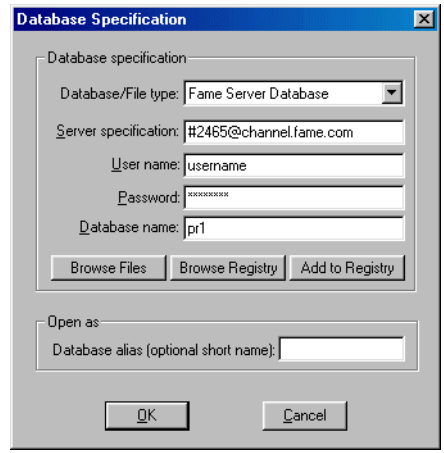
On the other hand, keeping the data in the foreign format may allow you to move between a number of applications without having to retranslate the file. This minimizes the number of copies of the data you have available, which may make the data easier to update and maintain.

Using EViews, you can either translate your data or work with your data directly in the foreign format. You should choose between the two based on your particular needs.

Opening a Foreign Database

Working with foreign formats requires very little additional knowledge. To open a foreign database, simply select **File/Open/Database...** from the main menu to open the dialog. In the field **Database/File Type:** select the type of the foreign database or file you wish to open. If the database is a local file, you can then use the **Browse Files** button to locate the database in exactly the same way as for a native EViews database. You can create a new foreign format database by a similar procedure way using **File/New/Database...** from the main EViews menu.

If the database is accessed through a client-server model, selecting the dialog will change to show extra fields necessary for making the connection to the server. For example, when accessing a database located on a FAME server, the dialog will include fields for the FAME server, username and password.



Since access to a server requires many fields to be entered, you may wish to save this information as an entry in the database registry (see "The Database Registry" on page 275 for details).

There are special issues relating to working with DRIPro links. See "DRIPro Link" on page 291 for details.

You can also create and open foreign format files using the `dbopen` or `dbcreate` commands. You may either use an option to specify the foreign type explicitly, or let EViews determine the type using the file extension. See `dbopen` (p. 266) and `dbcreate` (p. 265) in the *Command and Programming Reference* for details.

Copying a Foreign Database

Once you have opened a window to a foreign database, you can copy the entire database into a new format using **Proc/Copy the Database** from the database menus. A dialog will appear which allows you to specify the type and other attributes of the new database you would like to create.

When performing a database copy to a new format, objects which cannot be copied due to incompatibility between formats will result in error messages in the EViews command window but will not halt the copying process. Upon completion, a message in the status line reports how many objects could not be copied.

Notes on Particular Formats

DRIPro Link

A DRIPro link is a special type of database which allows you to fetch data remotely over the internet from Global Insight's extensive collection of economic data. To access these features, you must have a valid DRIPro account with Global Insight. There are special issues involved with using DRIPro links, which are discussed in detail in "DRIPro Link" on page 291.

DRIBase Database

The DRIBase system is a client server system used by Global Insight to provide databases at the client site which can be kept current by remote updates. Customers can also use

DRIBase as a means of storing their own databases in an Sybase or Microsoft SQL Server system.

DRIBase access is only available in the Enterprise Edition of EViews.

In order to access DRIBase databases, the TSRT library from Global Insight must already be installed on the client machine. This will normally be done by Global Insight as part of the DRIBase installation procedure.

When working with DRIBase databases, the **Server specification** field should be set to contain the DRIBase database prefix, while the **Database name** field should contain the DRIBase bank name, including the leading "@" where appropriate. Note that these fields, as well as the **Username** and **Password** fields may be case sensitive, so make sure to preserve the case of any information given to you.

A DRIBase database has slightly different handling of frequencies than most other databases supported by EViews. See "Issues with DRI Frequencies" on page 303 for details. You should also read "Dealing with Illegal Names" on page 303 for a discussion of how DRI names are automatically remapped by EViews.

For further information on DRIBase, please contact Global Insight directly (http://www.globalinsight.com).

FAME

The FAME format is a binary format written by FAME database products. FAME provides a variety of products and services for working with time series data.

FAME access is only available in the Enterprise Edition of EViews.

In order to access FAME databases, a valid installation of FAME must already be available. EViews makes use of the FAME C HLI library, and will error unless the FAME .DLLs are correctly installed on the machine. EViews currently supports only version 8 of the FAME libraries.

A local FAME database can have any file extension, and EViews supports access to a FAME database with any name. However, because many commands in EViews use the file extension to automatically detect the file type, you will generally find it easier to work with FAME databases which have the default ".DB" extension.

EViews also allows access to FAME databases located on a FAME Database Server. When working with a FAME server, the **Server specification** should be given in the form:

```
#port_number@ip_address
```

For example, the server specification for access to a FAME/Channel database might appear as:

```
#2552@channel.fame.com
```

Access to a server will require a valid username and password for that server.

Please contact FAME directly (http://www.fame.com) for further information about the FAME database system and other FAME products.

Haver

The Haver database format is a binary format used by Haver Analytics when distributing data.

Haver access is only available in the Enterprise Edition of EViews.

The main difference between Haver databases and other file formats supported by EViews is that Haver databases are read-only. You cannot create your own database in Haver format, nor can you modify an existing database. EViews will error if you try to do so.

Please contact Haver Analytics (http://www.haver.com) directly for further information about Haver Analytics data products.

AREMOS TSD

The TSD format is a portable ASCII file format written by the AREMOS package. Although EViews already has some support for TSD files through the `tsdftech`, `tsdstore`, `tsdload` and `tsdsave` commands, working with the database directly gives you an intuitive graphical interface to the data, and allows you to move data directly in and out of an EViews database without having to move the data through a workfile (which may force the data to be converted to a single frequency).

GiveWin/PcGive

The GiveWin/PcGive format is a binary file format used by GiveWin, PcGive versions 7 and 8, and PcFiml.

There are two issues when working with GiveWin/PcGive files. The first is that EViews is case insensitive when working with object names, while GiveWin and PcGive are case sensitive. Because of this, if you intend to work with a file in both packages, you should avoid having two objects with names distinguished only by case. If your files do not follow this rule, EViews will only be able to read the last of the objects with the same name. Any early objects will be invisible.

The second issue concerns files with mixed frequency. The GiveWin/PcGive file format does support series of mixed frequency, and EViews will write to these files accordingly. However, GiveWin itself appears to only allow you to read series from one frequency at a time, and will ignore (with error messages) any series which do not conform to the chosen

frequency. Consequently, depending on your application, you may prefer to store series of only one frequency per GiveWin/PcGive file.

RATS 4.x

The RATS 4.x format is a binary format used by RATS Version 4 on all platforms.

The main issue to be aware of when working with RATS 4.x format files is that the ".RAT" extension is also used by RATS version 3 files. EViews will neither read from nor write to RATS files in this earlier format. If you try to use EViews to open one of these files, EViews will error, giving you a message that the file has a version number which is not supported.

To work with a RATS Version 3 file in EViews, you will first have to use RATS to translate the file to the Version 4 format. To convert a Version 3 file to a Version 4 file, simply load the file into RATS and modify it in some way. When you save the file, RATS will ask you whether you would like to translate the file into the new format. One simple way to modify the file without actually changing the data is to rename a series in the file to the name which it already has. For example, if we have a Version 3 file called "OLDFILE.RAT", we can convert to a Version 4 by first opening the file for editing in RATS:

```
dedit oldfile.rat
```

then listing the series contained in the file:

```
catalog
```

then renaming one of the series (say "X") to its existing name

```
rename x x
```

and finally saving the file

```
save
```

At this point, you will be prompted whether you would like to translate the file into the Version 4 format.

See the RATS documentation for details.

RATS Portable

The RATS portable format is an ASCII format which can be read and written by RATS. It is generally slower to work with than RATS native format, but the files are human readable and can be modified using a text editor.

You can read the contents of a RATS portable file into memory in RATS with the following commands:

```
open data filename.trl
data(format=portable) start end list_of_series
close data
```

To write what is currently in memory in RATS to a RATS portable file, use:

```
open copy filename.trl
copy(format=portable) start end list_of_series
close copy
```

See the RATS documentation for details.

TSP Portable

The TSP portable format is an ASCII format which can be read and written by copies of TSP on all platforms. The file consists of a translation of a TSP native databank (which typically have the extension ".TLB") into a TSP program which, when executed, will regenerate the databank on the new machine.

To create a TSP portable file from a TSP databank file, use the DBCOPY command from within TSP:

```
dbcopy databank_name
```

To translate a TSP portable file back into a TSP databank file, simply execute the TSP file as a TSP program.

Once the data is in TSP databank format, you can use the TSP command ,

```
in databank_name
```

to set the automatic search to use this databank and the TSP command,

```
out databank_name
```

to save any series which are created or modified back to the databank.

See the TSP documentation for details.

EcoWin

EcoWin database support provides online access to economic and financial market data from EcoWin. The EcoWin Economic and Financial databases contain global international macroeconomic and financial data from more than 100 countries and multinational aggregates. Additional databases provide access to equities information and detailed country-specific information on earnings estimates, equities, funds, fixed income, and macroeco-

nomics. For further information on EcoWin data and software, please contact EcoWin directly (http://www.ecowin.com).

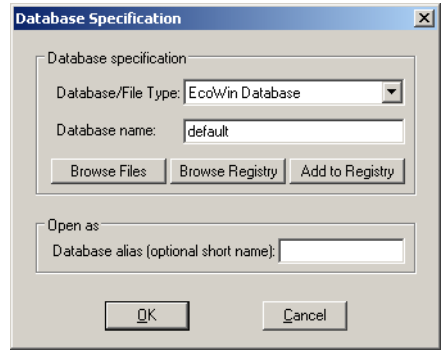
EcoWin database access is only available in the Enterprise Edition of EViews.

With EViews Enterprise Edition, you can open an EViews window into an online EcoWin database. This window allows browsing and text search of the series in the database, selecting series, and copying/exporting series into an EViews workfile or another EViews database. In addition, EViews provides a set of commands that may be used to perform tasks such as fetching a particular series from a EcoWin database.

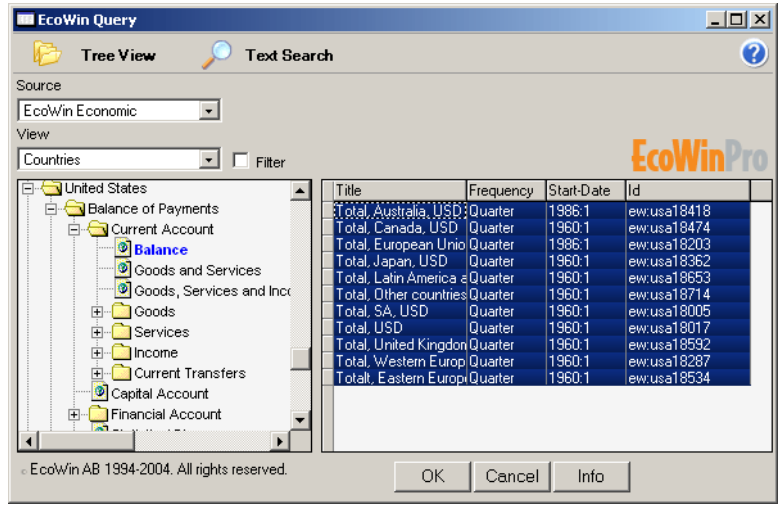
Access to EcoWin databases within EViews Enterprise Edition requires that the EcoWin Pro software has already been installed on the local machine, and that configuration of EcoWin database access using the EcoWin Database Configuration software has already been completed outside of EViews.

Interactive Graphical Interface

To open a graphical window to an EcoWin database, you should first open **Database Specification** dialog by selecting **File/Open/Database…**from the main EViews menu. Next, choose **EcoWin Database** in the **Database/File Type** combo, and enter the name of the online database as specified in the EcoWin Database Configuration software, typically "DEFAULT".

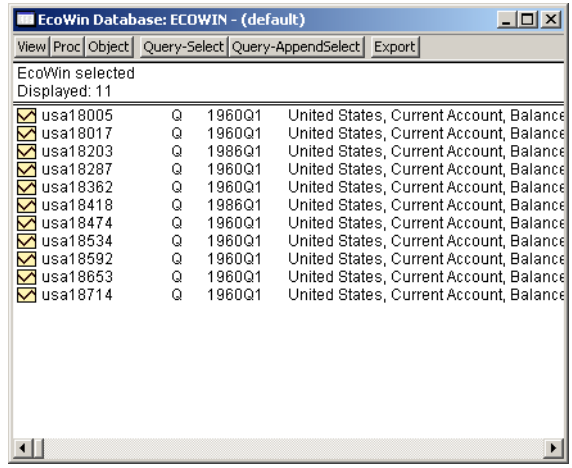


Clicking on **OK** will open an empty EViews database window. To access the EcoWin data, click on the **Query–Select** button in the database window toolbar. EViews will open a window containing a EcoWin Pro control for browsing and searching the online data. Note that may take a bit of time to initialize the EcoWin control. Once initialized, EViews will open the **EcoWin Query** window.



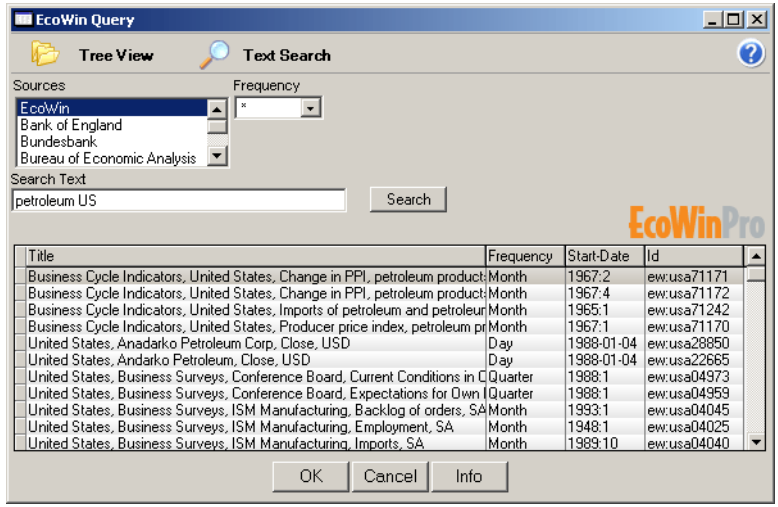
The **EcoWin Query** window provides you with two methods for selecting series to be brought into your EViews database.

First, you may use **Tree View** to browse a directory structure of the online database. You should use the tree on the left to navigate to the directory of interest, then select series in the window on the right by clicking or control-clicking on the entry, or by clicking on the right-mouse button and choosing **Select All**. Once the desired series have been highlighted, click on **OK** to bring the selected data into your EViews database.


EcoWin Database: ECOWIN - (default)
View
Proc
Object
Query-Select
Query-AppendSelect
Export
EcoWin selected
Displayed: 11
usa18005 Q 1960Q1 United States, Current Account, Balance
usa18017 Q 1960Q1 United States, Current Account, Balance
usa18203 Q 1986Q1 United States, Current Account, Balance
usa18287 Q 1960Q1 United States, Current Account, Balance
usa18362 Q 1960Q1 United States, Current Account, Balance
usa18418 Q 1986Q1 United States, Current Account, Balance
usa18474 Q 1960Q1 United States, Current Account, Balance
usa18534 Q 1960Q1 United States, Current Account, Balance
usa18592 Q 1960Q1 United States, Current Account, Balance
usa18653 Q 1960Q1 United States, Current Account, Balance
usa18714 Q 1960Q1 United States, Current Account, Balance

This procedure, first browsing to find a directory containing data of interest, selecting series, and then-clicking on **OK** to bring in data, can be performed multiple times, until a list of all the series that you wish to use has been accumulated within the EViews database window. At this point the EcoWin browse control can be closed using the **Cancel** button.

In place of browsing the tree structure of the database, you may elect to use text search to display a list of series in the database. Click on the **Text Search** selection at the top of the dialog to change the dialog to the search display, and enter the information in the appropriate fields. For example, to search for all series in the database using the text “PETROLEUM” and “US”, we have:


EcoWin Query
Tree View
Text Search
Sources
EcoWin
Bank of England
Bundesbank
Bureau of Economic Analysis
Frequency
Search Text
petroleum US
Search
EcoWinPro
Title
Frequency
Start-Date
Id
Business Cycle Indicators, United States, Change in PPI, petroleum product Month 1967:2 ew:usa71171
Business Cycle Indicators, United States, Change in PPI, petroleum product Month 1967:4 ew:usa71172
Business Cycle Indicators, United States, Imports of petroleum and petroleur Month 1965:1 ew:usa71242
Business Cycle Indicators, United States, Producer price index, petroleum pr Month 1967:1 ew:usa71170
United States, Anadarko Petroleum Corp, Close, USD Day 1988-01-04 ew:usa28850
United States, Andarko Petroleum, Close, USD Day 1988-01-04 ew:usa22665
United States, Business Surveys, Conference Board, Current Conditions in C Quarter 1988:1 ew:usa04973
United States, Business Surveys, Conference Board, Expectations for Own I Quarter 1988:1 ew:usa04959
United States, Business Surveys, ISM Manufacturing, Backlog of orders, SA Month 1993:1 ew:usa04045
United States, Business Surveys, ISM Manufacturing, Employment, SA Month 1948:1 ew:usa04025
United States, Business Surveys, ISM Manufacturing, Imports, SA Month 1989:10 ew:usa04040
OK
Cancel
Info

Highlight the series of interest and click on **OK** to bring them into the database. Repeat the tree browing or search method of adding series until the list in the database is complete, then click on **Cancel** to close the query window.

Once series of interest have been included in the database window, all of the standard EViews database tools, such as copy and paste into an existing workfile or database using the right mouse menus, creating a new EViews workfile containing the data using the **Export** button, or importing data into an existing EViews workfile using the **Fetch** menu item from the workfile window, are available.

Note that after you have completed your intitial query, you may reopen the EcoWin query window at any time. To add series to those already available in the database window, press the **Query Append Select** button in the database window, then browse or search for your series. To first clear the contents of the database window, you should press the **Query Select** button instead of the **Query Append Select** button.

Tips for Working with EcoWin Databases

If an EcoWin database is going to be used frequently or for direct access to individual series, you should find it useful to add an EcoWin entry in the database registry ("The Database Registry" on page 275).

The EViews database registry may be accessed by choosing **Options/Database Registry...** from the main EViews menu. Press **Add New Entry** to add a new database registry entry to the list. The procedure for adding an EcoWin database to the registry is identical to that for opening an EcoWin database. The **Database/File Type** field should be set to **EcoWin Data-**

base and the **Database Name/Path** field should be filled with the name assigned to the database in the EcoWin Database Configuration software (generally "DEFAULT").

Once the EcoWin database has been put in the registry, it may be referred to by its alias (short hand) name. For example, if you have assigned the EcoWin database the alias "EW", you can open the database with the simple command:

```
dbopen ew
```

or by using the **Browse Registry** button in the **Database Specification** dialog. The database name "EW" will be added to the most recently used file list, where it may be selected at a later time to reopen the database.

Assigning the EcoWin database a shorthand name also allows you to reference data without explicitly opening the database. For example, the command

```
equation eq1.ls ew::usa09016 c ew:usa09016(-1) @trend
```

runs a regression of U.S. unemployment on an intercept, its own lagged value, and a time trend. The series USA09016 will be accessed directly from the EcoWin servers, and does not need to appear within acurrently open database window for this command to be used. Other commands such as `copy` allow the name associated with the series to be changed during the procedure, as well as supporting the copying of series directly from an EcoWin database to another EViews database.

```
show ew::usa09016
```

displays a table of U. S. unemployment.

Note that series in the EcoWin "Economic" or EcoWin "Financial" databases may be referenced merely by using the database shorthand and the series name. In the example above, EViews looks for USA09016 in the two base EcoWin databases.

Series located in add-on EcoWin databases such as "Bank of England", "Bundesbank","Bureau of Economic Analysis", must also provide the name of the add-on database in which the series is located. You should provide the name of the EcoWin shortcut followed by a double colon, an EcoWin add-on database prefix, a slash, and then the series name. For example, you can fetch the mortgage rate (LUM5WTL) in the Bank of England database with

```
fetch ew::boe\lum5wtl
```

where we follow the datbase name with the add-on name BOE. The series will be named "BOE\LUM5WTL" in EViews. Note that the add-on name BOE is taken from the EcoWin name prefix (for example, LUM5WTL appears as "BOE:LUM5WTL" within EcoWin.

Working with DRIPro Links

EViews has the ability to remotely access databases hosted by Global Insight. Subscribers to Global Insight DRIPro data services can use these features to access data directly from within EViews.

Although the interface to remote databases is very similar to that of local databases, there are some differences due to the nature of the connection. There are also some issues specifically related to accessing DRI data. The following sections document these differences.

Enabling DRI Access

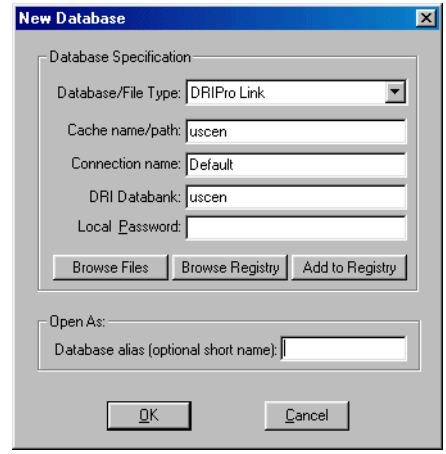
In order to access Global Insight data services, you will need to have an active DRIPro account. If you are not an existing DRIPro customer but may be interested in becoming one, you should contact Global Insight for details (http://www.globalinsight.com).

Access to DRI data will not be possible unless you have already installed and configured the DRIPro server software. If you have difficulties with getting the software to work, you should contact Global Insight directly for technical support.

Creating a Database Link

A remote Global Insight database is represented in EViews by a *database link*. A database link resembles a local database, consisting of a set of files on disk, but instead of containing the data itself, a database link contains information as to how to access the remote data. A database link also contains a cache in which copies of recently retrieved objects are kept, which can substantially reduce the time taken to perform some database operations.

You can create a database link by following a similar procedure to that used to create a local database. Select **File/New/Database…** from the main menu, then select **DRIPro Link** in the field **Database/File Type**. The dialog should change appearance so that a number of extra fields are displayed. Enter the name you would like to give the new database link in **Cache name/path**. You may wish to name the database link after the DRI databank to which it links.



In the **Connection name** field you should enter the name of the DRIPro connection you would like to use, as it appears in the **Connection Settings** box in the DRIPro configuration program. If you have only configured a single connection, and have not modified the connection name, the connection name will be DEFAULT, and this will be filled in automatically by EViews if you leave the field blank.

In the **DRI Databank** field you should input the full name of the DRIPro bank to which you would like to connect, not including any leading @ sign. For example, to connect to the DRI U.S. Central database, you should enter the name `uscen`. Each EViews database link may be associated with only one DRI databank, although you can create as many database links as you require.

The **Local Password** field may be used to set a password that must be entered whenever you wish to use the database link. This should not be confused with your DRIPro username and password, which you must already have provided in the DRIPro configuration program. Accessing a database link which contains a local password will cause a dialog to appear which prompts the user to input the password. Access to the remote database is only provided if the remote password is valid. Leave this field blank if you do not want a password to be attached to the database link.

When you have finished filling in the dialog fields, click on the **OK** button. A new database will be created and a database window should appear on the screen.

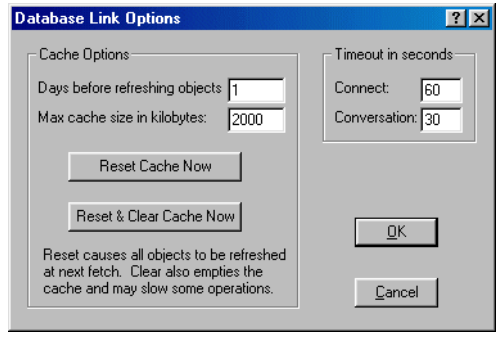
The database link window is very similar to a normal EViews database window. You should be able to perform basic query operations and simple fetching of series without any special instructions. Note, however, that it is not possible to modify a remote DRI database from within EViews, so operations which involve writing to the database have been removed. There are a number of other complications related to dealing with DRIPro databases that are described "Issues with DRI Frequencies" on page 303.

Understanding the Cache

A database link includes a cache of recently fetched objects which is used to speed up certain operations on the database. In some circumstances, fetching an object from the database will simply retrieve a copy from the local cache, rather than fetching a fresh copy of the data from the remote site. Even if a fresh copy is retrieved, having a previous copy of the series in the cache can substantially speed up retrieval.

You can regulate the caching behavior of the database link in a number of different ways. The basic option which determines under what circumstances a new copy of the data should be fetched is the *days before refresh*. If you attempt to fetch an object from the database link, and the copy of the object currently in the cache was fetched more recently than the days before refresh value, then the object currently in the cache will be returned instead of a fresh copy being fetched. For example, if days before refresh is set to one, any object which has already been fetched today will be retrieved from the cache, while any object which has not yet been fetched today will be retrieved from the remote site. Similarly, if days before refresh is set to seven, then an object in the cache must be more than a week old before a new copy of the object will be fetched. If days before refresh is set to zero, then a new copy of the data is fetched every time it is used.

You can change the days before refresh setting by clicking on the **Proc** button at the top of the database link window, then choosing **Link Options...** from the pop-up menu. A dialog will appear:



The dialog contains a number of fields, one of which is labeled **Days before refreshing objects**. Type a new number in the field to change the value.

The same dialog also contains a button marked **Reset cache now**. This button can be used to modify the behavior documented above. Clicking on the button causes the cache to mark all objects in the cache as out of date, so that the next time each object is fetched, it is guaranteed that a fresh copy will be retrieved. This provides a simple way for you to be certain that the database link will not return any data fetched before a particular time.

The dialog also contains some options for managing the size of the cache. The field marked **Maximum cache size in kilobytes** can be used to set the maximum size that the cache will be allowed to grow to on disk. If the cache grows above this size, a prompt will appear warning you that the cache has exceeded the limit and asking if you would like to compact the cache. Compacting is performed by deleting objects from oldest to newest until the cache size is reduced to less than three quarters of its maximum size. The cache is then packed to reclaim the empty space.

You can also completely clear the contents of the cache at any time by clicking on the button marked **Reset & Clear Cache Now**.

You can always examine the current contents of the database cache by clicking on the **Cache** button at the top of the database link window. This will display the names of all objects currently in the cache.

Configuring Link Options

The **Database Link Options** dialog also allows you to specify a number of timeout values. In most cases, the default values will behave acceptably. If you believe you are having problems with EViews aborting the connection too early, or you would like to shorten the times so as to receive a timeout message sooner, then enter new values in the appropriate fields.

- **Connection timeout—**is the length of time, in seconds, that EViews will wait for a response when first connecting to Global Insight. Depending on the type of connection you are making to Global Insight, this can take a significant amount of time.
- **Conversation timeout—**is the length of time, in seconds, that EViews will wait for a response from DRIPro when carrying out a transaction after a connection has already been made.

The values are attached to a particular database link, and can be reset at any time.

Dealing with Illegal Names

DRI databanks contain a number of series with names which are not legal names for EViews objects. In particular, DRI names frequently contain the symbols "@", "&" and "%", none of which are legal characters in EViews object names. We have provided a number of features to allow you to work with these series within EViews.

Because the "@" symbol is so common in DRI names, while the underline symbol (which is a legal character in EViews) is unused, we have hard-coded the rule that all underlines in EViews are mapped into "@" symbols in DRI names when performing operations on an DRI database link. For example, if there is a series with the name JQIMET@UK, you should refer to this series inside EViews as JQIMET_UK. Note that when performing queries, EViews will automatically replace the "@" symbol by an underline in the object name before displaying the query results on the screen. Consequently, if you are fetching data by copying-and-pasting objects from a query window, you do not need to be aware of this translation.

For other illegal names, you should use the object aliasing features (see "Object Aliases and Illegal Names" on page 285) to map the names into legal EViews object names.

Issues with DRI Frequencies

DRI databases have a different structure than EViews databases. An EViews database can contain series with mixed frequencies. A DRI database can contain data of only a single

frequency. In order that similar data may be grouped together, each DRI databank is actually composed of a series of separate databases, one for each frequency. When working with Global Insight data from within DRIPro software, you will often have to specify at exactly which frequency a particular series can be found. In some cases, a DRI databank may contain a series with the same name stored at several different frequencies.

Because this approach is inconsistent with the way that EViews works, we have tried to create a simpler interface to DRI data where you do not need to keep track of the frequency of each series that you would like to fetch. Instead, you can simply fetch a series by name or by selecting it from the query window, and EViews will do whatever is necessary to find out the frequency for you.

An ambiguity can arise in doing this, where a series with the same name appears at a variety of different frequencies in the DRI databank. By default, EViews resolves this ambiguity by always fetching the *highest* frequency data available. EViews will then perform necessary frequency conversions using the standard rules for frequency conversion in EViews (see "Frequency Conversion" on page 115).

In many cases, this procedure will exactly replicate the results that would be obtained if the lower frequency data was fetched directly from DRIPro. In some cases (typically when the series in question is some sort of ratio or other expression of one or more series), the figures may not match up exactly. In this case, if you know that the DRI data exists at multiple frequencies and you are familiar with DRI frequency naming conventions, you can explicitly fetch a series from a DRI database at a particular frequency by using a modified form of the command line form of fetch. Simply add the DRI frequency in parentheses after the name of the series. For example, the command:

```
fetch x(Q) y(A)
```

will fetch the series X and Y from the current default database, reading the quarterly frequency copy of X and the annual frequency copy of Y. If you request a frequency at which the data is not available, you will receive an error message. You should consult Global Insight documentation for details on DRI frequencies.

Limitations of DRI Queries

Queries to DRI database links are more limited than those available for EViews databases. The following section documents the restrictions.

First, queries on DRI databases allow only a subset of the fields available in EViews databases to be selected. The fields supported are: `name`, `type`, `freq`, `start`, `end`, `last_update` and `description`.

Second, the only fields which can be used in "where" conditions in a query on a DRIPro database link are name and description. (EViews does not support queries by frequency because of the ambiguities arising from DRI frequencies noted above).

Each of these fields has only one operator, the "matches" operator, and operations on the two fields can only be joined together using the "and" operator.

The "matches" operator is also limited for queries on DRI databases, matching only a subset of the expressions available for EViews databases. In particular, the pattern expression in a query on an DRI database must either have the form

```
a or b or … c
```

or the form

```
a and b and … c
```

Mixing of "and" and "or" is not allowed, and the "not" operator is not supported.

Patterns, however, are allowed and follow the normal EViews rules where "?" denotes any single character and "*" denotes zero or more characters.

Sorting of results by field is not supported.

Dealing with Common Problems

As stated in the introduction, you must install and configure the DRIPro software before EViews will be able to connect to Global Insight. If you cannot connect to Global Insight using the DRIPro software, you should contact Global Insight directly for assistance.

Assuming that you have correctly configured your DRIPro connection, in most cases EViews will be able to recover adequately from unexpected problems which arise during a DRIPro session without user intervention. Sometimes this will require EViews to automatically disconnect then reconnect to Global Insight.

There are some circumstances in which EViews may have problems making a connection. In order to connect to Global Insight, EViews uses a program written by Global Insight called DRIprosv. You can tell when this program is running by looking for the icon labeled "DRIpro server" in the Windows taskbar. Because of problems that can arise with multiple connections, EViews will not attempt to use the program if it is already running. Instead, EViews will report an error message "DRI server software already running". If there is another application which is using the connection to Global Insight, you can simply close down that program and the DRIPro server software should shut down automatically. If this is not the case, you may have to close down the DRIPro server software manually. Simply click on the icon in the Windows taskbar with the right mouse button, then select **Close** from the pop-up menu.

You may also use this as a procedure for forcing the DRIPro connection to terminate. Closing down the server software may cause EViews to report an error if it is currently carrying out a database transaction, but should otherwise be safe. EViews will restart the server software whenever it is needed.

Note that running other DRIPro software while EViews is using the DRIPro server software may cause EViews to behave unreliably.

Part II. Basic Data Analysis

The following chapters describe the EViews objects that you will use to perform basic data analysis.

- Chapter 11, "Series", beginning on page 309 describes the series object. Series are the basic unit of numeric data in EViews and are the basis for most univariate analysis. This chapter documents the basic data analysis and display features associated with series.
- Chapter 12, "Groups", on page 363 documents the views and procedures for the group object. Groups are collections of series (and like objects) which form the basis for a variety of multivariate graphical display and data analyses.
- Chapter 13, "Statistical Graphs from Series and Groups", on page 391 provides detailed documentation for exploratory data analysis using distribution graphs, kernel density, and scatterplot fit graphs.
- Chapter 14, "Graphs, Tables, and Text Objects", beginning on page 415 describes the creation and customization of tables and graph objects.

Chapter 11. Series

EViews provides various statistical graphs, descriptive statistics, and procedures as *views* and *procedures* of a numeric series. Once you have read or generated data into series objects using any of the methods described in Chapter 5, "Basic Data Handling", Chapter 6, "Working with Data", and Chapter 10, "EViews Databases", you are ready to perform statistical and graphical analysis using the data contained in the series.

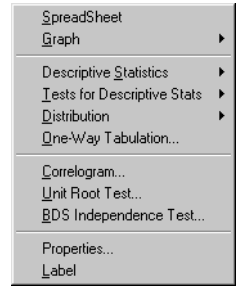
Series views compute various statistics for a single series and display these statistics in various forms such as spreadsheets, tables, and graphs. The views range from a simple line graph, to kernel density estimators. Series procedures create new series from the data in existing series. These procedures include various seasonal adjustment methods, exponential smoothing methods, and the Hodrick-Prescott filter.

The group object is used when working with more than one series at the same time. Methods which involve groups are described in Chapter 12, "Groups", on page 363.

To access the views and procedures for series, open the series window by double clicking on the series name in the workfile, or by typing `show` followed by the name of the series in the command window.

Series Views Overview

The series view drop-down menu is divided into four blocks. The first block lists views that display the underlying data in the series. The second and third blocks provide access to general statistics; the views in the third block are mainly for time series. The fourth block allows you to assign default series properties and to modify and display the series labels.



Spreadsheet and Graph Views

Spreadsheet

The spreadsheet view is the basic tabular view for the series data. Displays the raw, mapped, or transformed data series data in spreadsheet format.

You may customize your spreadsheet view extensively (see "Changing the Spreadsheet Display" in "Data Objects" on page 88).

In addition, the right-mouse button menu allows you to write the contents of the spreadsheet view to a CSV, tab-delimited ASCII text, RTF, or HTML file. Simply right-mouse button click, select the **Save table to disk...** menu item, and fill out the resulting dialog.

Graph

The graph submenu contains entries for various types of basic graphical display of the series: **Line**, **Area**, **Bar**, **Spike**, **Seasonal Stacked Line**, **Seasonal Split Line**.

- **Line** plots the series against the date/observation number.
- **Area** is a filled line graph.
- **Bar** plots the bar graph of the series. This view is useful for plotting series from a small data set that takes only a few distinct values.
- **Spike** plots a spike graph of the series against the date/observation number. The spike graph depicts values of the series as vertical spikes from the origin.
- **Seasonal Stacked Line** and **Seasonal Split Line** plot the series against observations reordered by season. The seasonal line graph view is currently available only for quarterly and monthly frequency workfiles.

  The stacked view reorders the series into seasonal groups where the first season observations are ordered by year, and then followed by the second season observations, and so on. Also depicted are the horizontal lines identifying the mean of the series in each season.

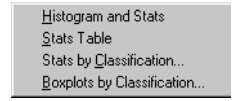
  The split view plots the line graph for each season on an annual horizontal axis.

See Chapter 14, “Graphs, Tables, and Text Objects”, beginning on page 415 for a discussion of techniques for modifying and customizing the graphical display.

Descriptive Statistics

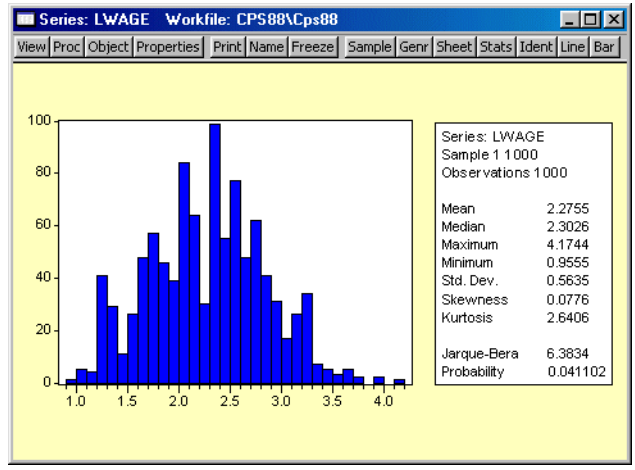
This set of views displays various summary statistics for the series. The submenu contains entries for histograms, basic statistics, statistics by classification, and boxplots by classification.

Histogram and Stats



This view displays the frequency distribution of your series in a histogram. The histogram divides the series range (the distance between the maximum and minimum values) into a number of equal length intervals or bins and displays a count of the number of observations that fall into each bin.

A complement of standard descriptive statistics are displayed along with the histogram. All of the statistics are calculated using the observations in the current sample.



- **Mean** is the average value of the series, obtained by adding up the series and dividing by the number of observations.
- **Median** is the middle value (or average of the two middle values) of the series when the values are ordered from the smallest to the largest. The median is a robust measure of the center of the distribution that is less sensitive to outliers than the mean.
- **Max** and **Min** are the maximum and minimum values of the series in the current sample.
- **Std. Dev.** (standard deviation) is a measure of dispersion or spread in the series. The standard deviation is given by:

$$s = \sqrt{\left(\sum_{i=1}^{N}(y_i - \bar{y})^2\right)/(N-1)} \tag{11.1}$$

  where N is the number of observations in the current sample and $\bar{y}$ is the mean of the series.
- **Skewness** is a measure of asymmetry of the distribution of the series around its mean. Skewness is computed as:

$$S = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \bar{y}}{\hat{\sigma}}\right)^3 \tag{11.2}$$

  where $\hat{\sigma}$ is an estimator for the standard deviation that is based on the biased estimator for the variance ($\hat{\sigma} = s\sqrt{(N-1)/N}$). The skewness of a symmetric distribution, such as the normal distribution, is zero. Positive skewness means that the distribution has a long right tail and negative skewness implies that the distribution has a long left tail.

- **Kurtosis** measures the peakedness or flatness of the distribution of the series. Kurtosis is computed as

$$K = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{y_i - \bar{y}}{\hat{\sigma}}\right)^4 \tag{11.3}$$

  where $\hat{\sigma}$ is again based on the biased estimator for the variance. The kurtosis of the normal distribution is 3. If the kurtosis exceeds 3, the distribution is peaked (leptokurtic) relative to the normal; if the kurtosis is less than 3, the distribution is flat (platykurtic) relative to the normal.

- **Jarque-Bera** is a test statistic for testing whether the series is normally distributed. The test statistic measures the difference of the skewness and kurtosis of the series with those from the normal distribution. The statistic is computed as:

$$\text{Jarque-Bera} = \frac{N-k}{6}\left(S^2 + \frac{(K-3)^2}{4}\right) \tag{11.4}$$

  where S is the skewness, K is the kurtosis, and k represents the number of estimated coefficients used to create the series.

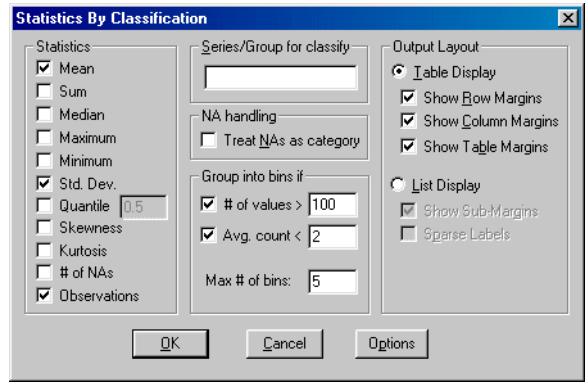
  Under the null hypothesis of a normal distribution, the Jarque-Bera statistic is distributed as χ^2 with 2 degrees of freedom. The reported Probability is the probability that a Jarque-Bera statistic exceeds (in absolute value) the observed value under the null hypothesis—a small probability value leads to the rejection of the null hypothesis of a normal distribution. For the LWAGE series displayed above, we reject the hypothesis of normal distribution at the 5% level but not at the 1% significance level.

Stats Table

Displays slightly more information than the **Histogram/Stats** view, with the numbers displayed in tabular form.

Stats by Classification

This view allows you to compute the descriptive statistics of a series for various subgroups of your sample. If you select **View/Descriptive Statistics/Stats by Classification…** a **Statistics by Classification** dialog box appears:



The **Statistics** option at the left allows you to choose the statistic(s) you wish to compute.

In the **Series/Group for Classify** field enter series or group names that define your subgroups. You must type at least one name. Descriptive statistics will be calculated for each unique value of the classification series unless binning is selected. You may type more than one series or group name; separate each name by a space. The quantile statistic requires an additional argument (a number between 0 and 1) corresponding to the desired quantile value. Click on the **Options** button to choose between various methods of computing the quantiles. See "CDF-Survivor-Quantile" on page 391 for details.

By default, EViews excludes observations which have missing values for any of the classification series. To treat NA values as a valid subgroup, select the **NA handling** option.

The **Layout** option allows you to control the display of the statistics. Table layout arrays the statistics in cells of two-way tables. The list form displays the statistics in a single line for each classification group.

The **Table** and **List** options are only relevant if you use more than one series as a classifier.

The **Sparse Labels** option suppresses repeating labels in list mode to make the display less cluttered.

The **Row Margins**, **Column Margins**, and **Table Margins** instruct EViews to compute statistics for aggregates of your subgroups. For example, if you classify your sample on the basis of gender and age, EViews will compute the statistics for each gender/age combination. If you elect to compute the marginal statistics, EViews will also compute statistics corresponding to each gender, and each age subgroup.

A classification may result in a large number of distinct values with very small cell sizes. By default, EViews automatically groups observations into categories to maintain moderate cell sizes and numbers of categories. **Group into Bins** provides you with control over this process.

Setting the **# of values** option tells EViews to group data if the classifier series takes more than the specified number of distinct values.

The **Avg. count** option is used to bin the series if the average count for each distinct value of the classifier series is less than the specified number.

The **Max # of bins** specifies the maximum number of subgroups to bin the series. Note that this number only provides you with approximate control over the number of bins.

The default setting is to bin the series into 5 subgroups if either the series takes more than 100 distinct values or if the average count is less than 2. If you do not want to bin the series, unmark both options.

For example, consider the following stats by classification view in table form:

Descriptive Statistics for LWAGE
Categorized by values of MARRIED and UNION
Date: 10/15/97 Time: 01:11
Sample: 1 1000
Included observations: 1000

Mean Median Std. Dev. Obs.		UNION 0	 1	 All
	0	1.993829 1.906575 0.574636 305	2.387019 2.409131 0.395838 54	2.052972 2.014903 0.568689 359
MARRIED	1	2.368924 2.327278 0.557405 479	2.492371 2.525729 0.380441 162	2.400123 2.397895 0.520910 641
	All	2.223001 2.197225 0.592757 784	2.466033 2.500525 0.386134 216	2.275496 2.302585 0.563464 1000

The header indicates that the table cells are categorized by two series MARRIED and UNION. These two series are dummy variables that take only two values. No binning is performed; if the series were binned, intervals rather than a number would be displayed in the margins.

The upper left cell of the table indicates the reported statistics in each cell; in this case, the median and the number of observations are reported in each cell. The row and column labeled **All** correspond to the **Row Margin** and **Column Margin** options described above.

Here is the same view in list form with sparse labels:

Descriptive Statistics for LWAGE
Categorized by values of MARRIED and UNION
Date: 10/15/97 Time: 01:08
Sample: 1 1000
Included observations: 1000

UNION	MARRIED	Mean	Median	Std. Dev.	Obs.
0	0	1.993829	1.906575	0.574636	305
	1	2.368924	2.327278	0.557405	479
	All	2.223001	2.197225	0.592757	784
1	0	2.387019	2.409131	0.395838	54
	1	2.492371	2.525729	0.380441	162
	All	2.466033	2.500525	0.386134	216
All	0	2.052972	2.014903	0.568689	359
	1	2.400123	2.397895	0.520910	641
	All	2.275496	2.302585	0.563464	1000

For series functions that compute by-group statistics, see "By-Group Statistics" on page 579 in the *Command and Programming Reference*.

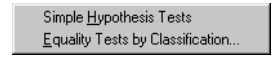
Boxplots by Classification

This view displays boxplots computed for various subgroups of your sample. For details, see "Boxplots" on page 409.

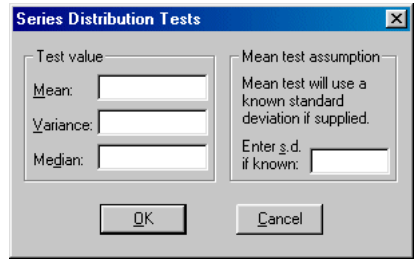
Tests for Descriptive Stats

This set of submenu entries contains views for performing hypothesis tests based on descriptive statistics for the series.

Simple Hypothesis Tests



This view carries out simple hypothesis tests regarding the mean, median, and the variance of the series. These are all single sample tests; see "Equality Tests by Classification" on page 318 for a description of two sample tests. If you select **View/Tests for Descriptive Stats/Simple Hypothesis Tests**, the Series Distribution Tests dialog box will be displayed.



Mean Test

Carries out the test of the null hypothesis that the mean μ of the series X is equal to a specified value m against the two-sided alternative that it is not equal to m:

$$H_0: \ \mu = m$$
$$H_1: \ \mu \neq m. \qquad (11.5)$$

If you do not specify the standard deviation of X, EViews reports a t-statistic computed as:

$$t = \frac{\overline{X} - m}{s/\sqrt{N}} \tag{11.6}$$

where $\overline{X}$ is the sample mean of X, s is the unbiased sample standard deviation, and N is the number of observations of X. If X is normally distributed, under the null hypothesis the t-statistic follows a t-distribution with $N-1$ degrees of freedom.

If you specify a value for the standard deviation of X, EViews also reports a z-statistic:

$$z = \frac{\overline{X} - m}{\sigma/\sqrt{N}} \tag{11.7}$$

where σ is the specified standard deviation of X. If X is normally distributed with standard deviation σ, under the null hypothesis, the z-statistic has a standard normal distribution.

To carry out the mean test, type in the value of the mean under the null hypothesis in the edit field next to **Mean**. If you want to compute the z-statistic conditional on a known standard deviation, also type in a value for the standard deviation in the right edit field. You can type in any number or standard EViews expression in the edit fields.

Hypothesis Testing for LWAGE
Date: 10/15/97 Time: 01:14
Sample: 1 1000
Included observations: 1000

Test of Hypothesis: Mean = 2

Sample Mean = 2.275496
Sample Std. Dev. = 0.563464

Method	Value	Probability
t-statistic	15.46139	0.00000

The reported probability value is the p-value, or marginal significance level, against a two-sided alternative. If this probability value is less than the size of the test, say 0.05, we reject the null hypothesis. Here, we strongly reject the null hypothesis for the two-sided test of equality. The probability value for a one-sided alternative is one half the p-value of the two-sided test.

Variance Test

Carries out the test of the null hypothesis that the variance of a series X is equal to a specified value σ^2 against the two-sided alternative that it is not equal to σ^2:

$$\begin{aligned} H_0&: \ \text{var}(x) = \sigma^2 \\ H_1&: \ \text{var}(x) \neq \sigma^2. \end{aligned} \tag{11.8}$$

EViews reports a χ^2 statistic computed as:

$$\chi^2 = \frac{(N-1)s^2}{\sigma^2} \tag{11.9}$$

where N is the number of observations, s is the sample standard deviation, and $\overline{X}$ is the sample mean of X. Under the null hypothesis and the assumption that X is normally distributed, the statistic follows a χ^2 distribution with $N-1$ degrees of freedom. The probability value is computed as $\min(p, 1-p)$, where p is the probability of observing a χ^2-statistic as large as the one actually observed under the null hypothesis.

To carry out the variance test, type in the value of the variance under the null hypothesis in the field box next to **Variance**. You can type in any *positive* number or expression in the field.

Median Test

Carries out the test of the null hypothesis that the median of a series X is equal to a specified value m against the two-sided alternative that it is not equal to m:

$$\begin{aligned} H_0&: \ \text{med}(x) = m \\ H_1&: \ \text{med}(x) \neq m. \end{aligned} \tag{11.10}$$

EViews reports three rank-based, nonparametric test statistics. The principal references for this material are Conover (1980) and Sheskin (1997).

- **Binomial sign test**. This test is based on the idea that if the sample is drawn randomly from a binomial distribution, the sample proportion above and below the true median should be one-half. Note that EViews reports two-sided p-values for both the sign test and the large sample normal approximation (with continuity correction).
- **Wilcoxon signed ranks test**. Suppose that we compute the absolute value of the difference between each observation and the mean, and then rank these observations from high to low. The Wilcoxon test is based on the idea that the sum of the ranks for the samples above and below the median should be similar. EViews reports a p-value for the asymptotic normal approximation to the Wilcoxon T-statistic (correcting for both continuity and ties). See Sheskin (1997, pp. 82–94) and Conover (1980, p. 284).
- **Van der Waerden (normal scores) test**. This test is based on the same general idea as the Wilcoxon test, but is based on smoothed ranks. The signed ranks are smoothed by converting them to quantiles of the normal distribution (normal scores). EViews reports the two-sided p-value for the asymptotic normal test described by Conover (1980).

To carry out the median test, type in the value of the median under the null hypothesis in the edit box next to **Median**. You can type any numeric expression in the edit field.

Hypothesis Testing for LWAGE
Date: 10/14/97 Time: 23:23
Sample: 1 1000
Included observations: 1000

Test of Hypothesis: Median = 2.25

Sample Median = 2.302585

Method	Value	Probability
Sign (exact binomial)	532	0.046291
Sign (normal approximation)	1.992235	0.046345
Wilcoxon signed rank	1.134568	0.256556
van der Waerden (normal scores)	1.345613	0.178427

Median Test Summary

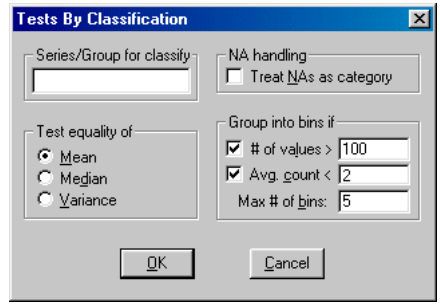
Category	Count	Mean Rank
Obs > 2.25	532	489.877820
Obs < 2.25	468	512.574786
Obs = 2.25	0	
Total	1000	

Equality Tests by Classification

This view allows you to test equality of the means, medians, and variances across subsamples (or subgroups) of a single series. For example, you can test whether mean income is the same for males and females, or whether the variance of education is related to race. The tests assume that the subsamples are independent.

For single sample tests, see the discussion of "Simple Hypothesis Tests" on page 315. For tests of equality across different series, see "Tests of Equality" on page 380.

Select **View/Tests for Descriptive Stats/Equality Tests by Classification...** and the **Tests by Classification** dialog box appears.



First, select whether you wish to test the mean, the median or the variance. Specify the subgroups, the NA handling, and the grouping options as described in "Stats by Classification" beginning on page 312.

Mean Equality Test

This test is based on a single-factor, between-subjects, analysis of variance (ANOVA). The basic idea is that if the subgroups have the same mean, then the variability between the

sample means (between groups) should be the same as the variability within any subgroup (within group).

Denote the i-th observation in group g as $x_{g,i}$, where $i = 1, \ldots, n_g$ for groups $g = 1, 2, \ldots G$. The between and within sums of squares are defined as:

$$SS_B = \sum_{g=1}^{G} n_g(\bar{x}_g - \bar{x})^2 \tag{11.11}$$

$$SS_B = \sum_{g=1}^{G} \sum_{i=1}^{n_g} (x_{ig} - \bar{x}_g)^2 \tag{11.12}$$

where $\bar{x}_g$ is the sample mean within group g and $\bar{x}$ is the overall sample mean. The F-statistic for the equality of means is computed as:

$$F = \frac{SS_B/(G-1)}{SS_W/(N-G)} \tag{11.13}$$

where N is the total number of observations. The F-statistic has an F-distribution with $G-1$ numerator degrees of freedom and $N-G$ denominator degrees of freedom under the null hypothesis of independent and identical normal distribution, with equal means and variances in each subgroup.

For tests with only two subgroups $(G = 2)$, EViews also reports the t-statistic, which is simply the square root of the F-statistic with one numerator degree of freedom. The top portion of the output contains the ANOVA results:

Test for Equality of Means of LWAGE
Categorized by values of MARRIED and UNION
Date: 02/24/04 Time: 12:09
Sample: 1 1000
Included observations: 1000

Method	df	Value	Probability
Anova F-statistic	(3, 996)	43.40185	0.0000

Analysis of Variance

Source of Variation	df	Sum of Sq.	Mean Sq.
Between	3	36.66990	12.22330
Within	996	280.5043	0.281631
Total	999	317.1742	0.317492

The analysis of variance table shows the decomposition of the total sum of squares into the between and within sum of squares, where:

Mean Sq. = Sum of Sq./df

The *F*-statistic is the ratio:

F = Between Mean Sq./Within Mean Sq.

The bottom portion of the output provides the category statistics:

Category Statistics

UNION	MARRIED	Count	Mean	Std. Dev.	Std. Err. of Mean
0	0	305	1.993829	0.574636	0.032904
0	1	479	2.368924	0.557405	0.025468
1	0	54	2.387019	0.395838	0.053867
1	1	162	2.492371	0.380441	0.029890
	All	1000	2.275496	0.563464	0.017818

Median (Distribution) Equality Tests

EViews computes various rank-based nonparametric tests of the hypothesis that the subgroups have the same general distribution, against the alternative that at least one subgroup has a different distribution.

In the two group setting, the null hypothesis is that the two subgroups are independent samples from the same general distribution. The alternative hypothesis may loosely be defined as "the values [of the first group] tend to differ from the values [of the second group]" (see Conover 1980, p. 281 for discussion). See also Bergmann, Ludbrook and Spooren (2000) for a more precise analysis of the issues involved.

We note that the "median" category in which we place these tests is somewhat misleading since the tests focus more generally on the equality of various statistics computed across subgroups. For example, the Wilcoxon test examines the comparability of mean ranks across subgroups. The categorization reflects common usage for these tests and various textbook definitions. The tests may, of course, have power against median differences.

- **Wilcoxon signed ranks test**. This test is computed when there are two subgroups. The test is identical to the Wilcoxon test outlined in the description of median tests ("Median Test" on page 317) but the division of the series into two groups is based upon the values of the classification variable instead of the value of the observation relative to the median.

- **Chi-square test** for the median. This is a rank-based ANOVA test based on the comparison of the number of observations above and below the overall median in each subgroup. This test is sometimes referred to as the *median test* (Conover, 1980).

  Under the null hypothesis, the median chi-square statistic is asymptotically distributed as a χ^2 with $G-1$ degrees of freedom. EViews also reports Yates' continuity corrected statistic. You should note that the use of this correction is controversial (Sheskin, 1997, p. 218).

- **Kruskal-Wallis one-way ANOVA by ranks**. This is a generalization of the Mann-Whitney test to more than two subgroups. The idea behind the Mann-Whitney test is to rank the series from smallest value (rank 1) to largest, and to compare the sum of the ranks from subgroup 1 to the sum of the ranks from subgroup 2. If the groups have the same median, the values should be similar.

  EViews reports the asymptotic normal approximation to the U-statistic (with continuity and tie correction) and the p-values for a two-sided test. For details, see Sheskin (1997). The test is based on a one-way analysis of variance using only ranks of the data. EViews reports the χ^2 chi-square approximation to the Kruskal-Wallis test statistic (with tie correction). Under the null hypothesis, this statistic is approximately distributed as a χ^2 with $G-1$ degrees of freedom (see Sheskin, 1997).

- **van der Waerden (normal scores) test**. This test is analogous to the Kruskal-Wallis test, except that we smooth the ranks by converting them into normal quantiles (Conover, 1980). EViews reports a statistic which is approximately distributed as a χ^2 with $G-1$ degrees of freedom under the null hypothesis. See the discussion of the Wilcoxon test for additional details on interpreting the test more generally as a test of a common subgroup distributions.

In addition to the test statistics and p-values, EViews reports values for the components of the test statistics for each subgroup of the sample. For example, the column labeled **Mean Score** contains the mean values of the van der Waerden scores (the smoothed ranks) for each subgroup.

Variance Equality Tests

Variance equality tests evaluate the null hypothesis that the variances in all G subgroups are equal against the alternative that at least one subgroup has a different variance. See Conover, *et al.* (1981) for a general discussion of variance testing.

- F**-test**. This test statistic is reported only for tests with two subgroups ($G=2$). First, compute the variance for each subgroup and denote the subgroup with the larger variance as L and the subgroup with the smaller variance as S. Then the F-statistic is given by:

$$F = s_L^2 / s_S^2 \tag{11.14}$$

where s_g^2 is the variance in subgroup g. This F-statistic has an F-distribution with $n_L - 1$ numerator degrees of freedom and $n_S - 1$ denominator degrees of freedom under the null hypothesis of equal variance and independent normal samples.

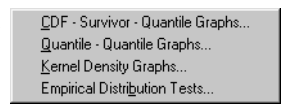
- **Siegel-Tukey test**. This test statistic is reported only for tests with two subgroups ($G = 2$). The test assumes the two subgroups are independent and have equal medians. The test statistic is computed using the same steps as the Kruskal-Wallis test described above for the median equality tests ("Median (Distribution) Equality Tests" on page 320), with a different assignment of ranks. The ranking for the Siegel-Tukey test alternates from the lowest to the highest value for every other rank. The Siegel-Tukey test first orders all observations from lowest to highest. Next, assign rank 1 to the lowest value, rank 2 to the *highest* value, rank 3 to the second highest value, rank 4 to the second lowest value, rank 5 to the third lowest value, and so on. EViews reports the normal approximation to the Siegel-Tukey statistic with a continuity correction (Sheskin, 1997, pp. 196–207).

- **Bartlett test**. This test compares the logarithm of the weighted average variance with the weighted sum of the logarithms of the variances. Under the joint null hypothesis that the subgroup variances are equal and that the sample is normally distributed, the test statistic is approximately distributed as a χ^2 with $G = 1$ degrees of freedom. Note, however, that the joint hypothesis implies that this test is sensitive to departures from normality. EViews reports the adjusted Bartlett statistic. For details, see Sokal and Rohlf (1995) and Judge, *et al.* (1985).

- **Levene test**. This test is based on an analysis of variance (ANOVA) of the absolute difference from the mean. The F-statistic for the Levene test has an approximate F-distribution with $G = 1$ numerator degrees of freedom and $N - G$ denominator degrees of freedom under the null hypothesis of equal variances in each subgroup (Levene, 1960).

- **Brown-Forsythe (modified Levene) test**. This is a modification of the Levene test in which we replace the absolute *mean* difference with the absolute *median* difference. The Brown-Forsythe test appears to be a superior in terms of robustness and power (Conover, *et al.* (1981), Brown and Forsythe (1974a, 1974b), Neter, *et al.* (1996)).

Distribution Graphs



These views display various graphs that characterize the empirical distribution of the series. A detailed description of these views may also be found in Chapter 13, "Statistical Graphs from Series and Groups", beginning on page 391.

CDF-Survivor-Quantile

This view plots the empirical cumulative distribution, survivor, and quantile functions of the series together with plus/minus two standard error bands. EViews provides a number of alternative methods for performing these computations.

Quantile-Quantile

The quantile-quantile (QQ)-plot is a simple yet powerful tool for comparing two distributions. This view plots the quantiles of the chosen series against the quantiles of another series or a theoretical distribution.

Kernel Density

This view plots the kernel density estimate of the distribution of the series. The simplest nonparametric density estimate of a distribution of a series is the histogram. The histogram, however, is sensitive to the choice of origin and is not continuous. The kernel density estimator replaces the "boxes" in a histogram by "bumps" that are smooth (Silverman 1986). Smoothing is done by putting less weight on observations that are further from the point being evaluated.

EViews provides a number of kernel choices as well as control over bandwidth selection and computational method.

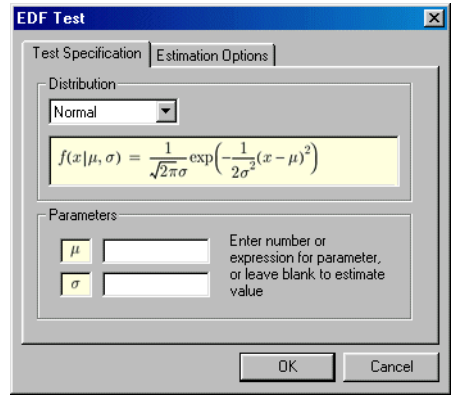
Empirical Distribution Tests

EViews provides built-in Kolmogorov-Smirnov, Lilliefors, Cramer-von Mises, Anderson-Darling, and Watson empirical distribution tests. These tests are based on the comparison between the empirical distribution and the specified theoretical distribution function. For a general description of empirical distribution function testing, see D'Agostino and Stephens (1986).

You can test whether your series is normally distributed, or whether it comes from, among others, an exponential, extreme value, logistic, chi-square, Weibull, or gamma distribution. You may provide parameters for the distribution, or EViews will estimate the parameters for you.

To carry out the test, simply double click on the series and select **View/Distribution/ Empirical Distribution Tests...** from the series window.

There are two tabs in the dialog. The **Test Specification** tab allows you to specify the parametric distribution against which you want to test the empirical distribution of the series. Simply select the distribution of interest from the drop-down menu. The small display window will change to show you the parameterization of the specified distribution.



You can specify the values of any known parameters in the edit field or fields. If you leave any field blank, EViews will estimate the corresponding parameter using the data contained in the series.

The **Estimation Options** tab provides control over any iterative estimation that is required. You should not need to use this tab unless the output indicates failure in the estimation process. Most of the options in this tab should be self-explanatory. If you select **User-specified starting values**, EViews will take the starting values from the C coefficient vector.

It is worth noting that some distributions have positive probability on a restricted domain. If the series data take values outside this domain, EViews will report an out-of-range error. Similarly, some of the distributions have restrictions on domain of the parameter values. If you specify a parameter value that does not satisfy this restriction, EViews will report an error message.

The output from this view consists of two parts. The first part displays the test statistics and associated probability values.

Empirical Distribution Test for DPOW2
Hypothesis: Normal
Date: 01/09/01 Time: 09:11
Sample: 1 1000
Included observations: 1000

Method	Value	Adj. Value	Probability
Lilliefors (D)	0.294098	NA	0.0000
Cramer-von Mises (W2)	27.89617	27.91012	0.0000
Watson (U2)	25.31586	25.32852	0.0000
Anderson-Darling (A2)	143.6455	143.7536	0.0000

Here, we show the output from a test for normality where both the mean and the variance are estimated from the series data. The first column, "Value", reports the asymptotic test statistics while the second column, "Adj. Value", reports test statistics that have a finite

sample correction or adjusted for parameter uncertainty (in case the parameters are estimated). The third column reports p-value for the adjusted statistics.

All of the reported EViews p-values will account for the fact that parameters in the distribution have been estimated. In cases where estimation of parameters is involved, the distributions of the goodness-of-fit statistics are non-standard and distribution dependent, so that EViews may report a subset of tests and/or only a range of p-value. In this case, for example, EViews reports the Lilliefors test statistic instead of the Kolmogorov statistic since the parameters of the normal have been estimated. Details on the computation of the test statistics and the associated p-values may be found in Anderson and Darling (1952, 1954), Lewis (1961), Durbin (1970), Dallal and Wilkinson (1986), Davis and Stephens (1989), Csörgö and Faraway (1996) and Stephens (1986).

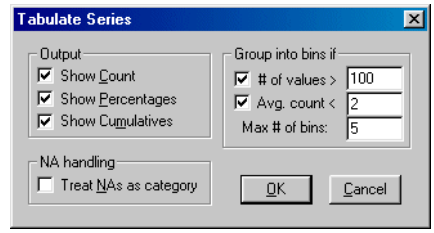
Method: Maximum Likelihood - d.f. corrected (Exact Solution)

Parameter	Value	Std. Error	z-Statistic	Prob.
MU	0.142836	0.015703	9.096128	0.0000
SIGMA	0.496570	0.011109	44.69899	0.0000
Log likelihood	-718.4084	Mean dependent var.		0.142836
No. of Coefficients	2	S.D. dependent var.		0.496570

The second part of the output table displays the parameter values used to compute the theoretical distribution function. Any parameters that are specified to estimate are estimated by maximum likelihood (for the normal distribution, the estimate of the standard deviation is degree of freedom corrected if the mean is not specified *a priori*). For parameters that do not have a closed form analytic solution, the likelihood function is maximized using analytic first and second derivatives. These estimated parameters are reported with a standard error and p-value based on the asymptotic normal distribution.

One-Way Tabulation

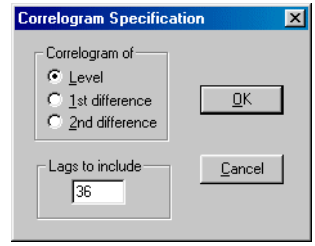
This view tabulates the series in ascending order, optionally displaying the counts, percentage counts, and cumulative counts. When you select **View/One-Way Tabulation...** the Tabulate Series dialog box will be displayed.



The Output options control which statistics to display in the table. You should specify the NA handling and the grouping options as described above in the discussion of "Stats by Classification" on page 312.

Cross-tabulation (n -way tabulation) is also available as a group view. See "N-Way Tabulation" on page 381 for details.

Correlogram



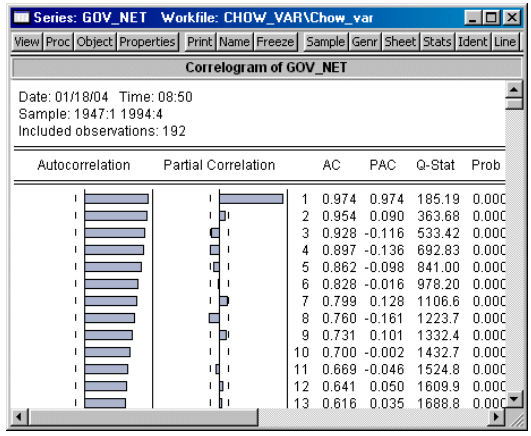
This view displays the autocorrelation and partial autocorrelation functions up to the specified order of lags. These functions characterize the pattern of temporal dependence in the series and typically make sense only for time series data. When you select **View/Correlogram...** the Correlogram Specification dialog box appears.

You may choose to plot the correlogram of the raw series (level) `x`, the first difference `d(x)=x-x(-1)`, or the second difference

```
d(x)-d(x(-1)) = x-2x(-1)+x(-2)
```

of the series.

You should also specify the highest order of lag to display the correlogram; type in a positive integer in the field box. The series view displays the correlogram and associated statistics:



Autocorrelations (AC)

The autocorrelation of a series Y at lag k is estimated by:

$$\tau_k = \frac{\sum_{t=k+1}^{T} (Y_t - \bar{Y})(Y_{t-k} - \bar{Y})}{\sum_{t=1}^{T} (Y_t - \bar{Y})^2} \tag{11.15}$$

where $\bar{Y}$ is the sample mean of Y. This is the correlation coefficient for values of the series k periods apart. If τ_1 is nonzero, it means that the series is first order serially correlated. If τ_k dies off more or less geometrically with increasing lag k, it is a sign that the series obeys a low-order autoregressive (AR) process. If τ_k drops to zero after a small number of lags, it is a sign that the series obeys a low-order moving-average (MA) process. See "Serial Correlation Theory" on page 493 for a more complete description of AR and MA processes.

Note that the autocorrelations estimated by EViews differ slightly from theoretical descriptions of the estimator:

$$\tau_k = \frac{\sum_{t=k+1}^{T} ((Y_t - \bar{Y})(Y_{t-k} - \bar{Y}))/(T-K)}{\sum_{t=1}^{T} (Y_t - \bar{Y})^2/T} \tag{11.16}$$

where $\bar{Y}_{t-k} = \sum Y_{t-k}/(T-k)$. The difference arises since, for computational simplicity, EViews employs the same overall sample mean $\bar{Y}$ as the mean of both Y_t and Y_{t-k}. While both formulations are consistent estimators, the EViews formulation biases the result toward zero in finite samples.

The dotted lines in the plots of the autocorrelations are the approximate two standard error bounds computed as $\pm 2/(\sqrt{T})$. If the autocorrelation is within these bounds, it is not significantly different from zero at (approximately) the 5% significance level.

Partial Autocorrelations (PAC)

The partial autocorrelation at lag k is the regression coefficient on Y_{t-k} when Y_t is regressed on a constant, Y_{t-1}, ..., Y_{t-k}. This is a *partial* correlation since it measures the correlation of Y values that are k periods apart after removing the correlation from the intervening lags. If the pattern of autocorrelation is one that can be captured by an autoregression of order less than k, then the partial autocorrelation at lag k will be close to zero.

The PAC of a pure autoregressive process of order p, AR(p), cuts off at lag p, while the PAC of a pure moving average (MA) process asymptotes gradually to zero.

EViews estimates the partial autocorrelation at lag k recursively by

$$\phi_k = \begin{cases} \tau_1 & \text{for } k = 1 \\ \dfrac{\tau_k - \sum_{j=1}^{k-1} \phi_{k-1,j}\,\tau_{k-j}}{1 - \sum_{j=1}^{k-1} \phi_{k-1,j}\,\tau_{k-j}} & \text{for } k > 1 \end{cases} \tag{11.17}$$

where τ_k is the estimated autocorrelation at lag k and where,

$$\phi_{k,j} = \phi_{k-1,j} - \phi_k \phi_{k-1,k-j}. \tag{11.18}$$

This is a consistent approximation of the partial autocorrelation. The algorithm is described in Box and Jenkins (1976, Part V, Description of computer programs). To obtain a more precise estimate of ϕ, simply run the regression:

$$Y_t = \beta_0 + \beta_1 Y_{t-1} + \ldots + \beta_{k-1} Y_{t-(k-1)} + \phi_k Y_{t-k} + e_t \tag{11.19}$$

where e_t is a residual. The dotted lines in the plots of the partial autocorrelations are the approximate two standard error bounds computed as $\pm 2/(\sqrt{T})$. If the partial autocorrelation is within these bounds, it is not significantly different from zero at (approximately) the 5% significance level.

Q-Statistics

The last two columns reported in the correlogram are the Ljung-Box Q-statistics and their p-values. The Q-statistic at lag k is a test statistic for the null hypothesis that there is no autocorrelation up to order k and is computed as:

$$Q_{LB} = T(T+2) \sum_{j=1}^{k} \frac{\tau_j^2}{T-J} \tag{11.20}$$

where τ_j is the j-th autocorrelation and T is the number of observations. If the series is not based upon the results of ARIMA estimation, then under the null hypothesis, Q is asymptotically distributed as a χ^2 with degrees of freedom equal to the number of autocorrelations. If the series represents the residuals from ARIMA estimation, the appropriate degrees of freedom should be adjusted to represent the number of autocorrelations less the number of AR and MA terms previously estimated. Note also that some care should be taken in interpreting the results of a Ljung-Box test applied to the residuals from an ARMAX specification (see Dezhbaksh, 1990, for simulation evidence on the finite sample performance of the test in this setting).

The Q-statistic is often used as a test of whether the series is white noise. There remains the practical problem of choosing the order of lag to use for the test. If you choose too small a lag, the test may not detect serial correlation at high-order lags. However, if you

choose too large a lag, the test may have low power since the significant correlation at one lag may be diluted by insignificant correlations at other lags. For further discussion, see Ljung and Box (1979) or Harvey (1990, 1993).

Unit Root Test

This view carries out the Augmented Dickey-Fuller (ADF), GLS transformed Dickey-Fuller (DFGLS), Phillips-Perron (PP), Kwiatkowski, *et. al.* (KPSS), Elliot, Richardson and Stock (ERS) Point Optimal, and Ng and Perron (NP) unit root tests for whether the series (or it's first or second difference) is stationary.

See "Nonstationary Time Series" on page 517 for a discussion of stationary and nonstationary time series and additional details on how to carry out the unit roots tests in Eviews.

BDS Test

This view carries out the BDS test for independence, as described in Brock, Dechert, Scheinkman and LeBaron (1996).

The BDS test is a portmanteau test for time based dependence in a series. It can be used for testing against a variety of possible deviations from independence including linear dependence, non-linear dependence, or chaos.

The test can be applied to a series of estimated residuals to check whether the residuals are independent and identically distributed (iid). For example, the residuals from an ARMA model can be tested to see if there is any non-linear dependence in the series after the linear ARMA model has been fitted.

The idea behind the test is fairly simple. To perform the test, we first choose a distance, ϵ. We then consider a pair of points. If the observations of the series truly are iid, then for any pair of points, the probability of the distance between these points being less than or equal to epsilon will be constant. We denote this probability by $c_1(\epsilon)$.

We can also consider sets consisting of multiple pairs of points. One way we can choose sets of pairs is to move through the consecutive observations of the sample in order. That is, given an observation s, and an observation t of a series X, we can construct a set of pairs of the form:

$$\{ \{X_s, X_t\}, \{X_{s+1}, X_{t+1}\}, \{X_{s+2}, X_{t+2}\}, \ldots, \{X_{s+m-1}, X_{t+m-1}\} \} \qquad (11.21)$$

where m is the number of consecutive points used in the set, or *embedding dimension*. We denote the joint probability of every pair of points in the set satisfying the epsilon condition by the probability $c_m(\epsilon)$.

The BDS test proceeds by noting that under the assumption of independence, this probability will simply be the product of the individual probabilities for each pair. That is, if the observations are independent,

$$c_m(\epsilon) = c_1{}^m(\epsilon). \tag{11.22}$$

When working with sample data, we do not directly observe $c_1(\epsilon)$ or $c_m(\epsilon)$. We can only estimate them from the sample. As a result, we do not expect this relationship to hold exactly, but only with some error. The larger the error, the less likely it is that the error is caused by random sample variation. The BDS test provides a formal basis for judging the size of this error.

To estimate the probability for a particular dimension, we simply go through all the possible sets of that length that can be drawn from the sample and count the number of sets which satisfy the ϵ condition. The ratio of the number of sets satisfying the condition divided by the total number of sets provides the estimate of the probability. Given a sample of n observations of a series X, we can state this condition in mathematical notation,

$$c_{m,n}(\epsilon) = \frac{2}{(n-m+1)(n-m)} \sum_{s=1}^{n-m+1} \sum_{t=s+1}^{n-m+1} \prod_{j=0}^{m-1} I_\epsilon(X_{s+j}, X_{t+j}) \tag{11.23}$$

where I_ϵ is the indicator function:

$$I_\epsilon(x, y) = \begin{cases} 1 & \text{if } |x-y| \le \epsilon \\ 0 & \text{otherwise.} \end{cases} \tag{11.24}$$

Note that the statistics $c_{m,n}$ are often referred to as *correlation integrals*.

We can then use these sample estimates of the probabilities to construct a test statistic for independence:

$$b_{m,n}(\epsilon) = c_{m,n}(\epsilon) - c_{1,n-m+1}(\epsilon)^m \tag{11.25}$$

where the second term discards the last $m-1$ observations from the sample so that it is based on the same number of terms as the first statistic.

Under the assumption of independence, we would expect this statistic to be close to zero. In fact, it is shown in Brock et al. (1996) that

$$(\sqrt{n-m+1})\frac{b_{m,n}(\epsilon)}{\sigma_{m,n}(\epsilon)} \to N(0,1) \tag{11.26}$$

where

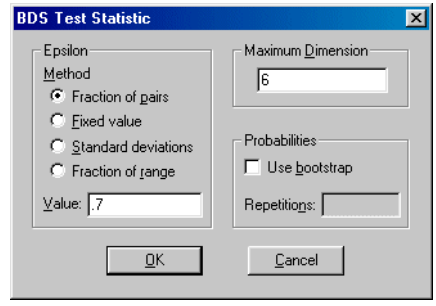
$$\sigma^2_{m,n}(\epsilon) = 4\left(k^m + 2\sum_{j=1}^{m-1} k^{m-j} c_1{}^{2j} + (m-1)^2 c_1{}^{2m} - m^2 k c_1{}^{2m-2}\right) \tag{11.27}$$

and where c_1 can be estimated using $c_{1,n}$. k is the probability of any triplet of points lying within ϵ of each other, and is estimated by counting the number of sets satisfying the sample condition:

$$k_n(\epsilon) = \frac{2}{n(n-1)(n-2)} \sum_{t=1}^{n} \sum_{s=t+1}^{n} \sum_{r=s+1}^{n} \tag{11.28}$$
$$(I_\epsilon(X_t, X_s)I_\epsilon(X_s, X_r) + I_\epsilon(X_t, X_r)I_\epsilon(X_r, X_s) + I_\epsilon(X_s, X_t)I_\epsilon(X_t, X_r))$$

To calculate the BDS test statistic in EViews, simply open the series you would like to test in a window, and choose **View/BDS Independence Test...**. A dialog will appear prompting you to input options.

To carry out the test, we must choose ϵ, the distance used for testing proximity of the data points, and the dimension m, the number of consecutive data points to include in the set.



The dialog provides several choices for how to specify ϵ:

- **Fraction of pairs**: ϵ is calculated so as to ensure a certain fraction of the total number of pairs of points in the sample lie within ϵ of each other.
- **Fixed value**: ϵ is fixed at a raw value specified in the units as the data series.
- **Standard deviations**: ϵ is calculated as a multiple of the standard deviation of the series.
- **Fraction of range**: ϵ is calculated as a fraction of the range (the difference between the maximum and minimum value) of the series.

The default is to specify ϵ as a fraction of pairs, since this method is most invariant to different distributions of the underlying series.

You must also specify the value used in calculating ϵ. The meaning of this value varies based on the choice of method. The default value of 0.7 provides a good starting point for the default method when testing shorter dimensions. For testing longer dimensions, you should generally increase the value of ϵ to improve the power of the test.

EViews also allows you to specify the maximum correlation dimension for which to calculate the test statistic. EViews will calculate the BDS test statistic for all dimensions from 2 to the specified value, using the same value of ϵ or each dimension. Note the same ϵ is

used only because of calculational efficiency. It may be better to vary ϵ with the correlation dimension to maximize the power of the test.

In small samples or in series that have unusual distributions, the distribution of the BDS test statistic can be quite different from the asymptotic normal distribution. To compensate for this, EViews offers you the option of calculating bootstrapped p-values for the test statistic. To request bootstrapped p-values, simply check the **Use bootstrap** box, then specify the number of repetitions in the field below. A greater number of repetitions will provide a more accurate estimate of the p-values, but the procedure will take longer to perform.

When bootstrapped p-values are requested, EViews first calculates the test statistic for the data in the order in which it appears in the sample. EViews then carries out a set of repetitions where for each repetition a set of observations is randomly drawn without replacement from the original data. Also note that the set of observations will be of the same size as the original data. For each repetition, EViews recalculates the BDS test statistic for the randomly drawn data, then compares the statistic to that obtained from the original data. When all the repetitions are complete, EViews forms the final estimate of the bootstrapped p-value by dividing the lesser of the number of repetitions above or below the original statistic by the total number of repetitions, then multiplying by two (to account for the two tails).

As an example of a series where the BDS statistic will reject independence, consider a series generated by the non-linear moving average model:

$$y_t = u_t + 8u_{t-1}u_{t-2} \tag{11.29}$$

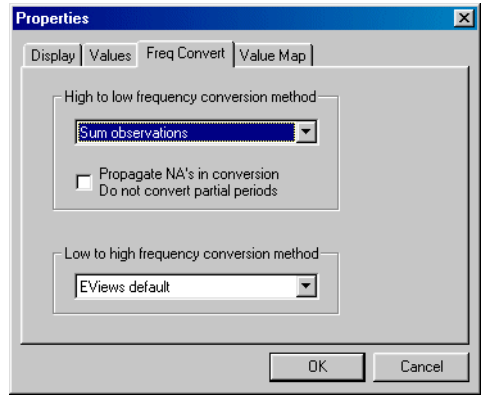
where u_t is a normal random variable. On simulated data, the correlogram of this series shows no statistically significant correlations, yet the BDS test strongly rejects the hypothesis that the observations of the series are independent (note that the Q-statistics on the squared levels of the series also reject independence).

Properties

Selecting **View/Properties...** provides access to the dialog controlling various series properties. Selecting this entry is equivalent to clicking on the **Properties** button on the series toolbar.

There are several tabs in the dialog. The first tab, labeled **Display**, allows you to set the default display characteristics for the series (see "Changing the Spreadsheet Display" on page 88). The **Values** tab may be used to define or modify a formula, turning the series into an auto-updating series, or to freeze the series values at their current levels (see "Defining an Auto-Updating Series" on page 150). The last **Value Map** tab should be used to assign value maps to the series (see "Value Maps" on page 163).

In dated workfiles, the **Freq Conversion** tab will also be displayed. You may use this tab to set the default frequency conversion settings for the series. Recall that when you fetch a series from an EViews database or when you copy a series to a workfile or workfile page with a different frequency, the series will automatically be converted to the frequency of the destination workfile. The conversion options view allows you to set the method that will be used to perform these conversions (see "Frequency Conversion" on page 115).

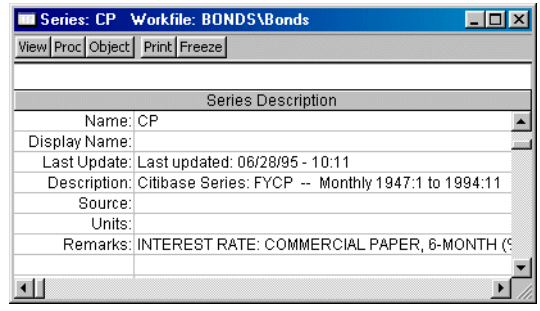


Each series has a default up and down frequency conversion method. By default, the series will take its settings from the EViews global options (see "Dates & Frequency Conversion" on page 939) in Appendix A, "Global Options", on page 937. This default series setting is labeled **EViews default**. You may, of course, override these settings for a given series. Here, instead of using the global defaults, the high to low conversion method is set to **Sum observations** without propagating NAs.

Label

This view displays a description of the series object.

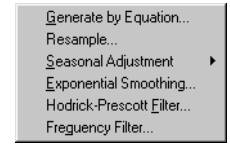
You can edit any of the field cells in the series label, except the **Last Update** cell which displays the date/time the series was last modified. Each field contains a single line, except for the **Remarks and History** fields which can contain up to 20 comment lines. Note that if you insert a line, the last (of the 20) line of these fields will be deleted.



The **Name** is the series name as it appears in the workfile; you can rename your series by editing this cell. If you fill in the **Display Name** field, this name may be used in tables and graphs in place of the standard object name. Unlike ordinary object names, **Display Names** may contain spaces and preserve capitalization (upper and lower case letters).

See Chapter 10, "EViews Databases", on page 261 for further discussion of label fields and their use in Database searches.

Series Procs Overview



Series procedures may be used to generate new series that are based upon the data in the original series.

When working with numeric series, you may use series procs to resample from the original series, to perform seasonal adjustment or exponential smoothing, or to filter the series using the Hodrick-Prescott or band-pass filters.

For alpha series you may, use a series proc to make a valmapped numeric series. EViews will create a new numeric series and valmap so that which each value in the numeric series is mapped to the original alpha series value.

Generate by Equation

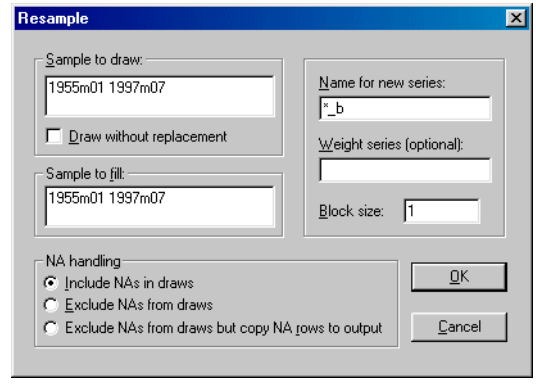
This is a general procedure that allows you to create new series by using expressions to transform the values in the existing series. The rules governing the generation of series are explained in detail in "Series Expressions" on page 131.

It is equivalent to using the `genr` command.

Resample

The series resampling procedure selects from the observations in a series to create a new series (the resampled series). You may draw your new sample with replacement (allow a given observation to be drawn multiple times) or without replacement. When you select **Proc/Resample...** from the series window, you will be prompted to specify various options.

Input Sample



Describes the sample from which observations are to be drawn. The default is the current workfile sample.

If you select the **Draw without replacement** option, each row will be drawn at most once. This option requires the input sample to be at least as large as the output sample. If you do not select this option, each row will be drawn *with* replacement.

Output Sample

Specifies the sample into which the resampled series will be saved. Any value outside the output sample will not be changed. The default output sample is the current workfile sample. If you select the **Draw without replacement** option, the output sample cannot be larger than the input sample.

NA Handling

The default **Include NAs in draws** instructs EViews to draw from every observation in the input sample, including those that contain missing values. Alternatively, you may select the **Exclude NAs from draws** option so that you draw only from observations in the input sample that do not contain any missing values. Finally, the **Exclude NAs from draws but copy NA rows to output** option first copies matching observations in the input sample that contain missing values to the output sample. The remaining rows of the output sample are then filled by drawing from observations in the input sample that do not contain any missing values. This option keeps observations with missing values fixed and resamples those that do not contain any missing values.

Series Name

The new series will be named using the specified series name. You may provide a series name or a wildcard expression. If you use a wildcard expression, EViews will substitute the existing series name in place of the wildcard. For example, if you are sampling from the series X and specify "*_SMP" as the output series, EViews will save the results in the series X_SMP. You may not specify a destination series that is the same as the original series.

If another series with the specified name exists in the workfile, the values in the output sample will be overwritten with the resampled values. Any values outside the output sample will remain unchanged. If there is a non-series object with the specified name, EViews will return an error message.

Because of these naming conventions, your original series cannot be an auto-series. For example, if the original series is X(-1) or LOG(X), EViews will issue an error. You will have to generate a new series, say by setting XLAG = X(-1) or LOGX = LOG(X), and then resample from the newly generated series.

Weighting

By default, the procedure draws from each row in the input sample with equal probabilities. If you want to attach different probabilities to the rows (importance sampling), you can specify a name of an existing series that contains weights that are proportional to the desired probabilities in each row. The weight series must have non-missing non-negative values in the input sample, but the weights need not add up to 1 since EViews will normalize the weights.

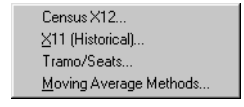
Block Length

By default, sets the *block length* to 1, meaning that we draw one observation at a time from the input sample. If you specify a block length larger than 1, EViews will draw blocks of consecutive rows of the specified length. The blocks drawn in the procedure form a set of *overlapping moving* blocks in the input sample. The drawn blocks will be appended one after the other in the output series until it fills the output sample (the final block will be truncated if the block size is not an integer multiple of the output sample size). Block resampling with a block length larger than 1 makes the most sense when resampling time series data.

Block resampling requires a continuous output sample. Therefore a block length larger than 1 cannot be used when the output sample contains "gaps" or when you have selected the **Exclude NAs from draws but copy NA rows to output** option. If you choose the **Exclude NAs from draws** option and the block length is larger than 1, the input sample will shrink in the presence of NAs in order to ensure that there are no missing values in any of the drawn blocks.

Seasonal Adjustment

Time series observed at quarterly and monthly frequencies often exhibit cyclical movements that recur every month or quarter. For example, ice cream sales may surge during summer every year and toy sales may reach a peak every December during Christmas sales. Seasonal adjustment refers to the process of removing these cyclical seasonal movements from a series and extracting the underlying trend component of the series.



The EViews seasonal adjustment procedures are available only for quarterly and monthly series. To seasonally adjust a series, click on **Proc/Seasonal Adjustment** in the series window toolbar and select the adjustment method from the submenu entries (**Census X11**, **X11 (Historical)**, **Tramo/Seats** or **Moving Average Methods**).

Census X12

EViews provides a convenient front-end for accessing the U.S. Census Bureau's X12 seasonal adjustment program from within EViews. The X12 seasonal adjustment program X12A.EXE is publicly provided by the Census and is installed in your EViews directory.

When you request X12 seasonal adjustment from EViews, EViews will perform all of the following steps:

- write out a specification file and data file for the series.
- execute the X12 program in the background, using the contents of the specification file.

- read back the output file and saved data into your EViews workfile.

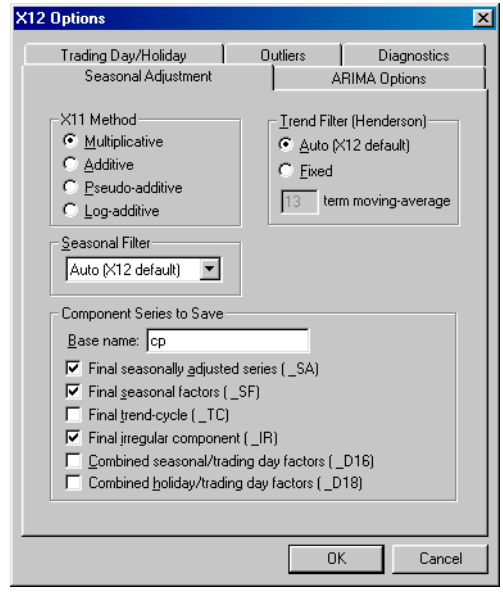
The following is a brief description of the EViews menu interface to X12. While some parts of X12 are not available via the menus, EViews also provides a more general command interface to the program (see `x12` (p. 550) of the *Command and Programming Reference*).

Users who desire a more detailed discussion of the X12 procedures and capabilities should consult the Census Bureau documentation. The full documentation for the Census program, *X12-ARIMA Reference Manual*, can be found in the DOCS subdirectory of your EViews directory in the PDF files (FINALPT1.PDF and FINALPT2.PDF).

To call the X12 seasonal adjustment procedure, select **Proc/Seasonal Adjustment/Census X12...** from the series window menu. A dialog will open with several tabs for setting the X12 options for seasonal adjustment, ARIMA estimation, trading day/holiday adjustment, outlier handling, and diagnostic output.

It is worth noting that when you open the X12 dialog, the options will be set to those from the previously executed X12 dialog. One exception to this rule is the outlier list in the Outliers tab, which will be cleared unless the previous seasonal adjustment was performed on the same series.

Seasonal Adjustment Options



X11 Method specifies the form of the seasonal adjustment decomposition. A description of the four choices can be found in pages 75-77 of the *X12-ARIMA Reference Manual*. Be aware that the Pseudo-additive method must be accompanied by an ARIMA specification (see "ARIMA Options" on page 339 for details on specifying the form of your ARIMA).

Note that the multiplicative, pseudo-additive, and log-additive methods do not allow for zero or negative data.

The **Seasonal Filter** drop-down box allows you to select a seasonal moving average filter to be used when estimating the seasonal factors. The default **Auto (X12 default)** setting is an automatic procedure based on the moving seasonality ratio. For details on the remaining seasonal filters, consult the X12-ARIMA Reference Manual. To approximate the results from the previous X11 program's default filter, choose the X11-default option. You should note the following:

- The seasonal filter specified in the dialog is used for all frequencies. If you wish to apply different filters to different frequencies, you will have to use the more general X12 command language described in detail in x12 (p. 550) of the *Command and Programming Reference*.
- X12 will not allow you to specify a 3×15 seasonal filter for series shorter than 20 years.
- The Census Bureau has confirmed that the X11-default filter option does not produce results which match those obtained from the previous version of X11. The difference arises due to changes in extreme value identification, replacement for the latest values, and the way the end weights of the Henderson filter is calculated. For comparability, we have retained the previous (historical) X11 routines as a separate procedure (see "Census X11 (Historical)" on page 344). Please note that the old X11 program is year 2000 compliant only through 2100 and supports only DOS 8.3 format filenames.

The **Trend Filter (Henderson)** settings allow you to specify the number of terms in the Henderson moving average used when estimating the trend-cycle component. You may use any odd number greater than 1 and less than or equal to 101. The default is the automatic procedure used by X12.

You must provide a base name for the series stored from the X12 procedure in the **Name for Adjusted Series/Component Series to Save** edit box. To save a series returned from X12 in the workfile, click on the appropriate check box. The saved series will have the indicated suffix appended to the base name. For example, if you enter a base name of "X" and ask to save the seasonal factors ("_SF"), EViews will save the seasonal factors as X_SF.

You should take care when using long base names, since EViews must be able to create a valid series using the base name and any appended Census designations. In interactive mode, EViews will warn you that the resulting name exceeds the maximum series name length; in batch mode, EViews will create a name using a truncated base name and appended Census designations.

The dialog only allows you to store the four most commonly used series. You may, however, store any additional series as listed on Table 6-8 (p. 74) of the *X12-ARIMA Reference Manual* by running X12 from the command line (see x12 (p. 550) of the *Command and Programming Reference*).

ARIMA Options

The X12 program also allows you to fit ARMA models to the series *prior* to seasonal adjustment. You can use X12 to remove deterministic effects (such as holiday and trading day effects) prior to seasonal adjustment and to obtain forecasts/backcasts that can be used for

seasonal adjustment at the boundary of the sample. To fit an ARMA, select the **ARIMA Options** tab in the **X12 Options** dialog and fill in the desired options.

The **Data Transformation** setting allows you to transform the series before fitting an ARMA model. The **Auto** option selects between no transformation and a log transformation based on the Akaike information criterion. The **Logistic** option transforms the series y to $\log(y/(1-y))$ and is defined only for series with values that are strictly between 0 and 1. For the **Box-Cox** option, you must provide the parameter value λ for the transformation

$$\begin{cases} \log(y_t) & \text{if } \lambda = 0 \\ \lambda^2 + (y_t^{\lambda} - 1)/\lambda & \text{if } \lambda \neq 0 \end{cases} \tag{11.30}$$

See the "transform spec" (pp. 60–67) of the *X12-ARIMA Reference Manual* for further details.

ARIMA Specification allows you to choose between two different methods for specifying your ARIMA model. The **Specify in-line** option asks you to provide a single ARIMA specification to fit. The X12 syntax for the ARIMA specification is different from the one used by EViews and follows the Box-Jenkins notation "(p d q)(P D Q)" where:

p	nonseasonal AR order
d	order of nonseasonal differences
q	nonseasonal MA order
P	(multiplicative) seasonal AR order
D	order of seasonal differences
Q	(multiplicative) seasonal MA order

The default specification "(0 1 1)(0 1 1)" is the seasonal IMA model:

$$(1-L)(1-L^s)y_t = (1-\theta_1 L)(1-\theta_s L^s)\epsilon_t \tag{11.31}$$

Here are some other examples (L is the lag operator):

(1 0 0)	$(1-\phi L)y_t = \epsilon_t$
(0 1 1)	$(1-L)y_t = (1-\theta L)\epsilon_t$
(1 0 1)(1 0 0)	$(1-\phi_1 L)(1-\phi_s L^s)y_t = (1-\theta L)\epsilon_t$ where $s = 4$ for quarterly data and $s = 12$ for monthly data.

You can skip lags using square brackets and explicitly specify the seasonal order after the parentheses:

([2 3] 0 0)	$(1-\phi_2 L^2-\phi_3 L^3)y_t = \epsilon_t$
(0 1 1)12	$(1-L^{12})y_t = (1-\theta L^{12})\epsilon_t$

See the *X12-ARIMA Reference Manual* (pp. 110–114) for further details and examples of ARIMA specification in X12. Note that there is a limit of 25 total AR, MA, and differencing coefficients in a model and that the maximum lag of any AR or MA parameter is 24 and the maximum number of differences in any ARIMA factor is 3.

Alternatively, if you choose **Select from file**, X12 will select an ARIMA model from a set of possible specifications provided in an external file. The selection process is based on a procedure developed by Statistics Canada for X11-ARIMA/88 and is described in the *X12-ARIMA Reference Manual* (p. 133). If you use this option, you will be asked to provide the name of a file that contains a set of possible ARIMA specifications. By default, EViews will use a file named X12A.MDL that contains a set of default specifications provided by Census (the list of specifications contained in this file is given below).

To provide your own list in a file, the ARIMA specification must follow the X12 syntax as explained in the ARIMA Specification section above. You must specify each model on a separate line, with an "X" at the end of each line except the last. You may also designate one of the models as a "default" model by marking the end of a line with an asterisk "*" instead of "X"; see p. 133 of the *X12-ARIMA Reference Manual* for an explanation of the use of a default model. To ensure that the last line is read, it should be terminated by hitting the return key.

For example, the default file (X12A.MDL) provided by X12 contains the following specifications:

```
(0 1 1)(0 1 1) *
(0 1 2)(0 1 1) x
(2 1 0)(0 1 1) x
(0 2 2)(0 1 1) x
(2 1 2)(0 1 1)
```

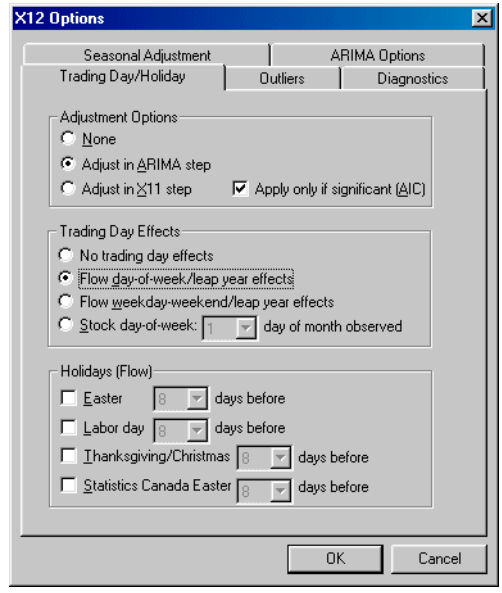
There are two additional options for **Select from file**. **Select best** checks all models in the list and looks for the model with minimum forecast error; the default is to select the first model that satisfies the model selection criteria. **Select by out-of-sample-fit** uses out-of-sample forecast errors (by leaving out some of the observations in the sample) for model evaluation; the default is to use within-sample forecast errors.

The **Regressors** option allows you to include prespecified sets of exogenous regressors in your ARIMA model. Simply use the checkboxes to specify a constant term and/or (centered) seasonal dummy variables. Additional predefined regressors to capture trading day

and/or holiday effects may be specified using the **Trading Day/Holiday** tab. You can also use the **Outlier** tab to capture outlier effects.

Trading Day and Holiday Effects

X12 provides options for handling trading day and/or holiday effects. To access these options, select the **Trading Day/Holiday** tab in the **X12 Options** dialog.



As a first step you should indicate whether you wish to make these adjustments in the ARIMA step or in the X11 seasonal adjustment step. To understand the distinction, note that there are two main procedures in the X12 program: the X11 seasonal adjustment step, and the ARIMA estimation step. The X11 step itself consists of several steps that decompose the series into the trend/cycle/irregular components. The X12 procedure may therefore be described as follows:

- optional preliminary X11 step (remove trading day/holiday effects from series, if requested).
- ARIMA step: fit an ARIMA model (with trading/holiday effects, if specified) to the series from step 1 or to the raw series.
- X11 step: seasonally adjust the series from step 2 using backcasts/forecasts from the ARIMA model.

While it is possible to perform trading day/holiday adjustments in *both* the X11 step and the ARIMA step, Census recommends against doing so (with a preference to performing the adjustment in the ARIMA step). EViews follows this advice by allowing you to perform the adjustment in only one of the two steps.

If you choose to perform the adjustment in the X11 step, there is an additional setting to consider. The checkbox **Apply only if significant (AIC)** instructs EViews to adjust only if warranted by examination of the Akaike information criterion.

It is worth noting that in X11, the significance tests for use of trading day/holiday adjustment are based on an F-test. For this, and a variety of other reasons the X12 procedure with "X11 settings" will not produce results that match those obtained from historical X11.

To obtain comparable results, you must use the historical X11 procedure (see "Census X11 (Historical)" on page 344).

Once you select your adjustment method, the dialog will present additional adjustment options:

- **Trading Day Effects** — There are two options for trading day effects, depending on whether the series is a flow series or a stock series (such as inventories). For a flow series, you may adjust for day-of-week effects or only for weekday-weekend contrasts. Trading day effects for stock series are available only for monthly series and the day of the month in which the series is observed must be provided.
- **Holiday Effects** — Holiday effect adjustments apply only to flow series. For each holiday effect, you must provide a number that specifies the duration of that effect *prior* to the holiday. For example, if you select 8, the level of daily activity changes on the seventh day *before* the holiday and remains at the new level until the holiday (or a day before the holiday, depending on the holiday).

Note that the holidays are as defined for the United States and may not apply to other countries. For further details, see the *X12-ARIMA Reference Manual*, Tables 6–15 (p. 94) and 6–18 (p. 133).

Outlier Effects

As with trading day/holiday adjustments, outlier effects can be adjusted either in the X11 step or in the ARIMA step (see the discussion in "Trading Day and Holiday Effects" on page 341). However, outlier adjustments in the X11 step are done only to robustify the trading day/holiday adjustments in the X11 step. Therefore, in order to perform outlier adjustment in the X11 step, you must perform trading day/holiday adjustment in the X11 step. Only additive outliers are allowed in the X11 step; other types of outliers are available in the ARIMA step. For further information on the various types of outliers, see the *X12-ARIMA Reference Manual*, Tables 6–15 (p. 94) and 6–18 (p. 133).

If you do not know the exact date of an outlier, you may ask the program to test for an outlier using the built-in X12 diagnostics.

Diagnostics

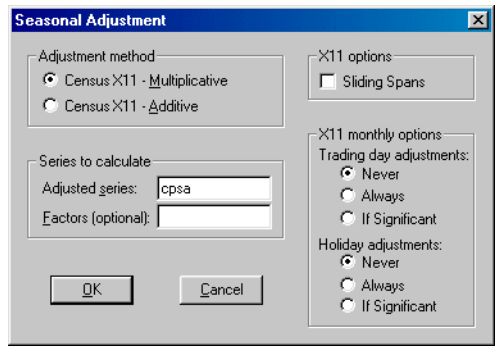
This tab provides options for various diagnostics. The **Sliding spans** and **Historical revisions** options test for stability of the adjusted series. While **Sliding spans** checks the change in adjusted series over a moving sample of fixed size (overlapping subspans), **Historical revisions** checks the change in adjusted series over an increasing sample as new observations are added to the sample. See the *X12-ARIMA Reference Manual* for further details and references of the testing procedure. You may also choose to display various diagnostic output:

- **Residual diagnostics** will report standard residual diagnostics (such as the autocorrelation functions and Q-statistics). These diagnostics may be used to assess the adequacy of the fitted ARIMA model. Note that this option requires estimation of an ARIMA model; if you do not provide an ARIMA model nor any exogenous regressors (including those from the **Trading day/Holiday** or **Outlier** tab), the diagnostics will be applied to the original series.
- **Outlier detection** automatically detects and reports outliers using the specified ARIMA model. This option requires an ARIMA specification or at least one exogenous regressor (including those from the **Trading day/Holiday** or **Outlier** tab); if no regression model is specified, the option is ignored.
- **Spectral plots** displays the spectra of the differenced seasonally adjusted series (SP1) and/or of the outlier modified irregular series (SP2). The red vertical dotted lines are the seasonal frequencies and the black vertical dashed lines are the trading day frequencies. If you observe peaks at these vertical lines it is an indication of inadequate adjustment. For further details, see Findley *et al.* (1998, section 3.1). If you request this option, data for the spectra will be stored in a matrix named *seriesname*_SA_SP1 and *seriesname*_SA_SP2 in your workfile. The first column of these matrices are the frequencies and the second column are 10 times the log spectra at the corresponding frequency.

X11/X12 Troubleshooting

The currently shipping versions of X11 and X12 as distributed by the Census have the following limitation regarding directory length. First, you will not be able to run X11/X12 if you are running EViews from a shared directory on a server which has spaces in its name. The solution is to map that directory to a letter drive on your local machine. Second, the temporary directory path used by EViews to read and write data cannot have more than four subdirectories. This temporary directory used by EViews can be changed by selecting **Options/File Locations.../Temp File Path** in the main menu. If your temporary directory has more than four subdirectories, change the Temp File Path to a writeable path that has fewer subdirectories. Note that if the path contains spaces or has more than 8 characters, it may appear in shortened form compatible with the old DOS convention.

Census X11 (Historical)



The Census X11.2 methods (multiplicative and additive) are the standard methods used by the U.S. Bureau of Census to seasonally adjust publicly released data. The X11 routines are separate programs provided by the Census and are installed in the EViews directory in the files X11Q2.EXE and X11SS.EXE. The documentation for these programs can also be found in your EViews directory as text files X11DOC1.TXT through X11DOC3.TXT.

The X11 programs may be executed directly from DOS or from within EViews. If you run the X11 programs from within EViews, the adjusted series and the factor series will be automatically imported into your EViews workfile. X11 summary output and error messages will also be displayed in the series window at the end of the procedure.

The X11 method has many options, the most important of which are available in the Seasonal Adjustment dialog. However, there are other options not available in the EViews dialog; to use these other options, you should run the X11 programs from the DOS command line. All options available in the X11 methods are described in the X11DOC text files in your EViews directory.

You should note that there is a limit on the number of observations that you can seasonally adjust. X11 only works for quarterly and monthly frequencies, requires at least four full years of data, and can adjust only up to 20 years of monthly data and up to 30 years of quarterly data.

Tramo/Seats

Tramo ("Time Series Regression with ARIMA Noise, Missing Observations, and Outliers") performs estimation, forecasting, and interpolation of regression models with missing observations and ARIMA errors, in the presence of possibly several types of outliers. *Seats* ("Signal Extraction in ARIMA Time Series") performs an ARIMA-based decomposition of an observed time series into unobserved components. The two programs were developed by Victor Gomez and Agustin Maravall.

Used together, Tramo and Seats provide a commonly used alternative to the Census X12 program for seasonally adjusting a series. Typically, individuals will first "linearize" a series using Tramo and will then decompose the linearized series using Seats.

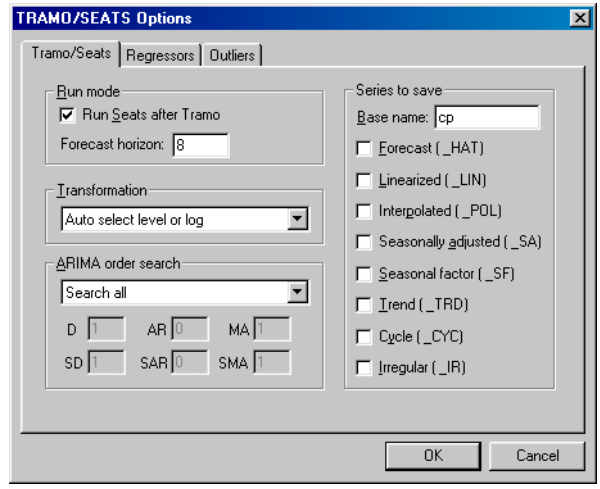
EViews provides a convenient front-end to the Tramo/Seats programs as a series proc. Simply select **Proc/Seasonal Adjustment/Tramo Seats...** and fill out the dialog. EViews writes an input file which is passed to Tramo/Seats via a call to a .DLL, and reads the output files from Tramo/Seats back into EViews (note: since EViews uses a new .DLL version of Tramo/Seats, results may differ from the older DOS version of the program).

Since EViews only provides an interface to an external program, we cannot provide any technical details or support for Tramo/Seats itself. Users who are interested in the technical details should consult the original documentation *Instructions for the User* which is provided as a .PDF file in the DOCS/TRAMOSEATS subdirectory of your EViews directory.

Dialog Options

The Tramo/Seats interface from the dialog provides access to the most frequently used options. Users who desire more control over the execution of Tramo/Seats may use the command line form of the procedure as documented in `tramoseats` (p. 512) in the *Command and Programming Reference*.

The dialog contains three tabs. The main tab controls the basic specification of your Tramo/Seats run.



- **Run mode**: You can choose either to run only Tramo or you can select the **Run Seats after Tramo** checkbox to run both. In the latter case, EViews uses the input file produced by Tramo to run Seats. If you wish to run only Seats, you must use the command line interface.

- **Forecast horizon**: You may set the number of periods to forecast outside the current sample. If you choose a number smaller than the number of forecasts required to run Seats, Tramo will automatically lengthen the forecast horizon as required.

- **Transformation**: Tramo/Seats is based on an ARIMA model of the series. You may choose to fit the ARIMA model to the level of the series or to the (natural) log of the series, or you select **Auto select level or log**. This option automatically chooses between the level model and the log transformed model using results from a

trimmed range-mean regression; see the original Tramo/Seats documentation for further details.

- **ARIMA order search**: You may either specify the orders of the ARIMA model to fit or ask Tramo to search for the "best" ARIMA model. If you select **Fix order** in the combo box and specify the order of all of the ARIMA components, Tramo will use the specified values for all components where the implied ARIMA model is of the form:

$$\begin{aligned} y_t &= x_t'\beta + u_t \\ \phi(L)\delta(L)u_t &= \theta(L)\epsilon_t \\ \delta(L) &= (1-L)^D(1-L^s)^{SD} \\ \phi(L) &= (1+\phi_1 L + \ldots + \phi_{AR}L^{AR})(1+\Phi_1 L^s + \ldots + \Phi_{SAR}(L^s)^{SAR}) \\ \theta(L) &= (1+\theta_1 L + \ldots + \theta_{MA}L^{MA})(1+\Theta_1 L^s + \ldots + \Theta_{SMA}(L^s)^{SMA}) \end{aligned}$$

with seasonal frequency s. When you fix the order of your ARIMA you should specify non-negative integers in the edit fields for D, SD, AR, SAR, MA, and SMA.

Alternatively, if you select **Fix only difference orders**, Tramo will search for the best ARMA model for differenced data of the orders specified in the edit fields.

You can also instruct Tramo to choose all orders. Simply choose **Search all** or **Search all and unit complex roots** to have Tramo find the best ARIMA model subject to limitations imposed by Tramo. The two options differ in the handling of complex roots. Details are provided in the original Tramo/Seats documentation.

Warning: if you choose to run Seats after Tramo, note that Seats has the following limit on the ARIMA orders: $D \le 3$, $AR \le 3$, $MA \le 3$, $SD \le 2$, $SAR \le 1$, $SMA \le 1$.

- **Series to Save**: To save series output by Tramo/Seats in your workfile, provide a valid base name and check the series you wish to save. The saved series will have a postfix appended to the basename as indicated in the dialog. If the saved series contains only missing values, it indicates that Tramo/Seats did not return the requested series; see "Trouble Shooting" on page 349.

If Tramo/Seats returns forecasts for the selected series, EViews will append them at the end of the stored series. The workfile *range* must have enough observations after the current workfile *sample* to store these forecasts.

If you need access to series that are not listed in the dialog options, see "Trouble Shooting" on page 349.

- **User specified exogenous series**: You may provide your own exogenous series to be used by Tramo. These must be a named series or a group in the current workfile and should not contain any missing values in the current sample and the forecast period.

  If you selected a trading day adjustment option, you have the option of specifying exogenous series to be treated as a holiday series. The specification of the holiday series will depend on whether you chose a weekday/weekend adjustment or a 5-day adjustment. See the original Tramo/Seats documentation for further details.

  If you are running Seats after Tramo, you must specify which component to allocate the regression effects. The Tramo default is to treat the regression effect as a separate additional component which is not included in the seasonally adjusted series.

  EViews will write a separate data file for each entry in the exogenous series list which is passed to Tramo. If you have many exogenous series with the same specification, it is best to put them into one group.

- **Easter/Trading day adjustment**: These options are intended for monthly data; see the original Tramo/Seats documentation for details.

- **Outlier detection**: You may either ask Tramo to automatically detect possible outliers or you can specify your own outlier but not both. If you wish to do both, create a series corresponding to the known outlier and pass it as an exogenous series.

  Similarly, the built-in intervention option in Tramo is not supported from the dialog. You may obtain the same result by creating the intervention series in EViews and passing it as an exogenous series. See the example below.

  The original Tramo/Seats documentation provides definitions of the various outlier types and the method to detect them.

After you click **OK**, the series window will display the text output returned by Tramo/Seats. If you ran both Tramo and Seats, the output from Seats is appended at the end of Tramo output. Note that this text view will be lost if you change the series view. You should freeze the view into a text object if you wish to refer to the output file without having to run Tramo/Seats again.

It is worth noting that when you run Tramo/Seats, the dialog will generally contain the settings from the previous run of Tramo/Seats. A possible exception is the user specified outlier list which is cleared unless Tramo/Seats is called on the previously used series.

Comparing X12 and Tramo/Seats

Both X12 and Tramo/Seats are seasonal adjustment procedures based on extracting components from a given series. Methodologically, X12 uses a non-parametric moving average based method to extract its components, while Tramo/Seats bases its decomposition on an estimated parametric ARIMA model (the recent addition of ARIMA modelling in X12

appears to be used mainly to identify outliers and to obtain backcasts and forecasts for end-of-sample problems encountered when applying moving average methods.)

For the practitioner, the main difference between the two methods is that X12 does not allow missing values while Tramo/Seats will interpolate the missing values (based on the estimated ARIMA model). While both handle quarterly and monthly data, Tramo/Seats also handles annual and semi-annual data. See the sample programs in the **Example Files** directory for a few results that compare X12 and Tramo/Seats.

Trouble Shooting

Error handling

As mentioned elsewhere, EViews writes an input file which is passed to Tramo/Seats via a call to a .DLL. Currently the Tramo/Seats .DLL does not return error codes. Therefore, the only way to tell that something went wrong is to examine the output file. If you get an error message indicating that the output file was not found, the first thing you should do is to check for errors in the input file.

When you call Tramo/Seats, EViews creates two subdirectories called Tramo and Seats in a temporary directory. This temporary directory is taken from the global option **Options/File Locations.../Temp File Path** (note that long directory names with spaces may appear in shortened DOS form). The Temp File Path can be retrieved in a program by a call to the function `@temppath` (p. 641) as described in the *Command and Programming Reference*.

The Tramo input file written by EViews will be placed in the subdirectory TRAMO and is named SERIE. A Seats input file written by Tramo is also placed in subdirectory TRAMO and is named SEATS.ITR.

The input file used by Seats is located in the SEATS subdirectory and is named SERIE2. If Seats is run alone, then EViews will create the SERIE2 file. When Tramo and Seats are called together, the Tramo file SEATS.ITR is copied into SERIE2.

If you encounter the error message containing the expression “output file not found”, it probably means that Tramo/Seats encountered an error in one of the input files. You should look for the input files SERIE and SERIE2 in your temp directories and check for any errors in these files.

Retrieving additional output

The output file displayed in the series window is placed in the OUTPUT subdirectory of the TRAMO and/or SEATS directories. The saved series are read from the files returned by Tramo/Seats that are placed in the GRAPH subdirectories. If you need to access other data files returned by Tramo/Seats that are not supported by EViews, you will have to read them back into the workfile using the `read` command from these GRAPH subdirectories. See the PDF documentation file for a description of these data file formats.

Warning: if you wish to examine these files, make sure to read these data files before you run the next Tramo/Seats procedure. EViews will clear these subdirectories before running the next Tramo/Seats command (this clearing is performed as a precautionary measure so that Tramo/Seats does not read results from a previous run).

Moving Average Methods

Ratio to moving average—multiplicative

The algorithm works as follows. Denote the series to be filtered by y_t.

1. First compute the centered moving average of y_t as:

$$x_t = \begin{cases} (0.5y_{t+6} + \ldots + y_t + \ldots 0.5y_{t-6})/12 & \text{if monthly} \\ (0.5y_{t+2} + y_{t+1} + y_t + y_{t-1} + 0.5y_{t-1})/4 & \text{if quarterly} \end{cases} \tag{11.32}$$

2. Take the ratio $\tau_t = y_t/x_t$.

3. Compute the seasonal indices. For monthly series, the seasonal index i_m for month m is the average of τ_t using observations only for month m. For quarterly series, the seasonal index i_q for quarter q is the average of τ_t using observations only for quarter q.

4. We then adjust the seasonal indices so that they multiply to one. This is done by computing the seasonal factors as the ratio of the seasonal index to the geometric mean of the indices:

$$s = \begin{cases} i_m/(\sqrt[12]{i_1 i_2 \ldots i_{12}}) & \text{if monthly} \\ i_q/(\sqrt[4]{i_1 i_2 i_3 i_4}) & \text{if quarterly} \end{cases} \tag{11.33}$$

5. These s are the reported *scaling factors* in the series window and are saved as series if you provide a name in the field box. The interpretation is that the series y is s_j percent higher in period j relative to the adjusted series.

6. The seasonally adjusted series is obtained by dividing y_t by the seasonal factors s_j.

Difference from moving average—additive

Suppose that we wish to filter y_t.

1. First compute the centered moving average of y_t as in Equation (11.32) on page 350.

2. Take the difference $d_t = y_t - x_t$.

3. Compute the seasonal indices. For monthly series, the seasonal index i_m for month m is the average of d_t using observations only for month m. For quarterly series, the seasonal index i_q for quarter q is the average of d_t using observations only for quarter q.

4. We then adjust the seasonal indices so that they add up to zero. This is done by setting $s_j = i_j - \bar{i}$ where $\bar{i}$ is the average of all seasonal indices. These s are the reported *scaling factors*. The interpretation is that the series y is s_j higher in period j relative to the adjusted series.

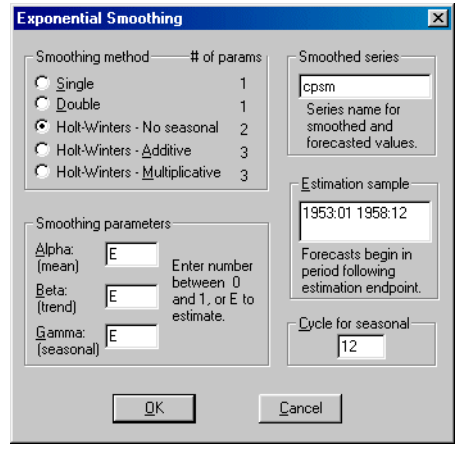
5. The seasonally adjusted series is obtained by subtracting the seasonal factors s_j from y_t.

The main difference between X11 and the moving average methods is that the seasonal factors may change from year to year in X11. The seasonal factors are assumed to be constant for the moving average method.

Exponential Smoothing

Exponential smoothing is a simple method of adaptive forecasting. It is an effective way of forecasting when you have only a few observations on which to base your forecast. Unlike forecasts from regression models which use fixed coefficients, forecasts from exponential smoothing methods adjust based upon past forecast errors. For additional discussion, see Bowerman and O'Connell (1979).

To obtain forecasts based on exponential smoothing methods, choose **Proc/Exponential Smoothing**. The Exponential Smoothing dialog box appears:



You need to provide the following information:

- **Smoothing Method**. You have the option to choose one of the five methods listed.

- **Smoothing Parameters**. You can either specify the values of the smoothing parameters or let EViews estimate them.

To estimate the parameter, type the letter e (for estimate) in the edit field. EViews estimates the parameters by minimizing the sum of squared errors. Don't be surprised if the

estimated damping parameters are close to one—it is a sign that the series is close to a random walk, where the most recent value is the best estimate of future values.

To specify a number, type the number in the field corresponding to the parameter. All parameters are constrained to be between 0 and 1; if you specify a number outside the unit interval, EViews will estimate the parameter.

- **Smoothed Series Name**. You should provide a name for the smoothed series. By default, EViews will generate a name by appending SM to the original series name, but you can enter any valid EViews name.
- **Estimation Sample**. You must specify the sample period upon which to base your forecasts (whether or not you choose to estimate the parameters). The default is the current workfile sample. EViews will calculate forecasts starting from the first observation after the end of the estimation sample.
- **Cycle for Seasonal**. You can change the number of seasons per year from the default of 12 for monthly or 4 for quarterly series. This option allows you to forecast from unusual data such as an undated workfile. Enter a number for the cycle in this field.

Single Smoothing (one parameter)

This single exponential smoothing method is appropriate for series that move randomly above and below a constant mean with no trend nor seasonal patterns. The smoothed series $\hat{y}_t$ of y_t is computed recursively, by evaluating:

$$\hat{y}_t = \alpha y_t + (1-\alpha)\hat{y}_{t-1} \tag{11.34}$$

where $0 < \alpha \leq 1$ is the *damping* (or *smoothing*) factor. The smaller is the α, the smoother is the $\hat{y}_t$ series. By repeated substitution, we can rewrite the recursion as

$$\hat{y}_t = \alpha \sum_{s=0}^{t-1} (1-\alpha)^s y_{t-s} \tag{11.35}$$

This shows why this method is called exponential smoothing—the forecast of y_t is a weighted average of the past values of y_t, where the weights decline exponentially with time.

The forecasts from single smoothing are constant for all future observations. This constant is given by:

$$\hat{y}_{T+k} = \hat{y}_T \qquad \text{for all } k > 0 \tag{11.36}$$

where T is the end of the estimation sample.

To start the recursion, we need an initial value for $\hat{y}_t$ and a value for α. EViews uses the mean of the initial observations of y_t to start the recursion. Bowerman and O'Connell

(1979) suggest that values of α around 0.01 to 0.30 work quite well. You can also let EViews estimate α to minimize the sum of squares of one-step forecast errors.

Double Smoothing (one parameter)

This method applies the single smoothing method twice (using the same parameter) and is appropriate for series with a linear trend. Double smoothing of a series y is defined by the recursions:

$$\begin{aligned} S_t &= \alpha y_t + (1-\alpha) S_{t-1} \\ D_t &= \alpha S_t + (1-\alpha) D_{t-1} \end{aligned} \tag{11.37}$$

where S is the single smoothed series and D is the double smoothed series. Note that double smoothing is a single parameter smoothing method with damping factor $0 < \alpha \leq 1$.

Forecasts from double smoothing are computed as:

$$\begin{aligned} \hat{y}_{T+k} &= \left(2 + \frac{\alpha k}{1-\alpha}\right) S_T - \left(1 + \frac{\alpha k}{1-\alpha}\right) D_T \\ &= \left(2S_T - D_T + \frac{\alpha}{1-\alpha}(S_T - D_T)k\right) \end{aligned} \tag{11.38}$$

The last expression shows that forecasts from double smoothing lie on a linear trend with intercept $2S_T - D_T$ and slope $\alpha(S_T - D_T)/(1-\alpha)$.

Holt-Winters—Multiplicative (three parameters)

This method is appropriate for series with a linear time trend and multiplicative seasonal variation. The smoothed series $\hat{y}_t$ is given by,

$$\hat{y}_{t+k} = (a + bk) c_{t+k} \tag{11.39}$$

where

a	permanent component (intercept)
b	trend
c_t	multiplicative seasonal factor

(11.40)

These three coefficients are defined by the following recursions:

$$\begin{aligned} a(t) &= \alpha \frac{y_t}{c_t(t-s)} + (1-\alpha)(a(t-1)+b(t-1)) \\ b(t) &= \beta(a(t)-a(t-1)) + (1-\beta)b(t-1) \\ c_t(t) &= \gamma \frac{y_t}{a(t)} + (1-\gamma)c_t(t-s) \end{aligned} \tag{11.41}$$

where $0 < \alpha, \beta, \gamma < 1$ are the damping factors and s is the seasonal frequency specified in the **Cycle for Seasonal** field box.

Forecasts are computed by:

$$\hat{y}_{t+k} = (a(T) + b(T)k)c_{T+k-s} \tag{11.42}$$

where the seasonal factors are used from the last s estimates.

Holt-Winters—Additive (three parameter)

This method is appropriate for series with a linear time trend and additive seasonal variation. The smoothed series $\hat{y}_t$ is given by:

$$\hat{y}_{t+k} = a + bk + c_{t+k} \tag{11.43}$$

where a and b are the permanent component and trend as defined above in Equation (11.40) and c are the additive seasonal factors. The three coefficients are defined by the following recursions:

$$\begin{aligned} a(t) &= \alpha(y_t - c_t(t-s)) + (1-\alpha)(a(t-1)+b(t-1)) \\ b(t) &= \beta(a(t)-a(t-1)) + 1 - \beta b(t-1) \\ c_t(t) &= \gamma(y_t - a(t+1)) - \gamma c_t(t-s) \end{aligned} \tag{11.44}$$

where $0 < \alpha, \beta, \gamma < 1$ are the damping factors and s is the seasonal frequency specified in the **Cycle for Seasonal** field box.

Forecasts are computed by:

$$\hat{y}_{T+k} = a(T) + b(T)k + c_{T+k-s} \tag{11.45}$$

where the seasonal factors are used from the last s estimates.

Holt-Winters—No Seasonal (two parameters)

This method is appropriate for series with a linear time trend and no seasonal variation. This method is similar to the double smoothing method in that both generate forecasts with a linear trend and no seasonal component. The double smoothing method is more parsimonious since it uses only one parameter, while this method is a two parameter method. The smoothed series $\hat{y}_t$ is given by:

$$\hat{y}_{t+k} = a + bk \tag{11.46}$$

where a and b are the permanent component and trend as defined above in Equation (11.40).

These two coefficients are defined by the following recursions:;

$$\begin{aligned} a(t) &= \alpha y_t + (1-\alpha)(a(t-1) + b(t-1)) \\ b(t) &= \beta(a(t) - a(t-1)) + 1 - \beta b(t-1) \end{aligned} \tag{11.47}$$

where $0 < \alpha, \beta, \gamma < 1$ are the damping factors. This is an exponential smoothing method with two parameters.

Forecasts are computed by:

$$\hat{y}_{T+k} = a(T) + b(T)k \tag{11.48}$$

These forecasts lie on a linear trend with intercept $a(T)$ and slope $b(T)$.

It is worth noting that Holt-Winters—No Seasonal, is *not* the same as additive or multiplicative with $\gamma = 0$. The condition $\gamma = 0$ only restricts the seasonal factors from changing over time so there are still (fixed) nonzero seasonal factors in the forecasts.

Illustration

As an illustration of forecasting using exponential smoothing we forecast data on monthly housing starts (HS) for the period 1985M01–1988M12 using the DRI Basics data for the period 1959M01–1984M12. These data are provided in the workfile HS.WF1. Load the workfile, highlight the HS series, double click, select **Proc/Exponential Smoothing…**. We use the Holt-Winters—multiplicative method to account for seasonality, name the smoothed forecasts as HS_SM, and estimate all parameters over the period 1959M1–1984M12.

When you click **OK**, EViews displays the results of the smoothing procedure. The first part displays the estimated (or specified) parameter values, the sum of squared residuals, the root mean squared error of the forecast. The zero values for Beta and Gamma in this example mean that the trend and seasonal components are estimated as fixed and not changing.

Date: 10/15/97 Time: 00:57
Sample: 1959:01 1984:12
Included observations: 312
Method: Holt-Winters Multiplicative Seasonal
Original Series: HS
Forecast Series: HS_SM

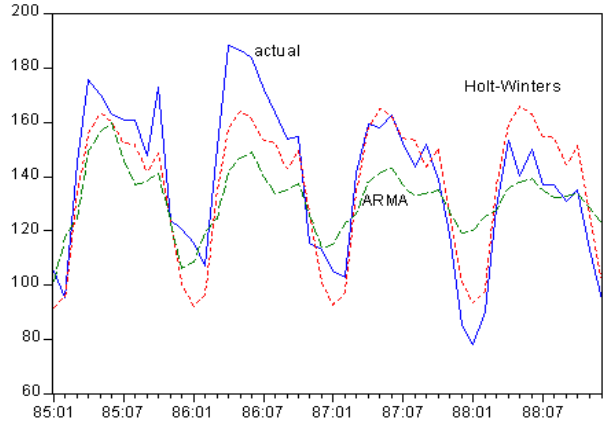
Parameters:	Alpha	0.7100
	Beta	0.0000
	Gamma	0.0000
Sum of Squared Residuals		40365.69
Root Mean Squared Error		11.37441

The second part of the table displays the mean (α), and trend (β) at the end of the estimation sample that are used for post-sample smoothed forecasts.

End of Period Levels:	Mean		134.6584
	Trend		0.064556
	Seasonals:	1984:01	0.680745
		1984:02	0.711559
		1984:03	0.992958
		1984:04	1.158501
		1984:05	1.210279
		1984:06	1.187010
		1984:07	1.127546
		1984:08	1.121792
		1984:09	1.050131
		1984:10	1.099288
		1984:11	0.918354
		1984:12	0.741837

For seasonal methods, the seasonal factors (γ) used in the forecasts are also displayed. The smoothed series in the workfile contains data from the beginning of the estimation sample to the end of the workfile range; all values after the estimation period are forecasts.

When we plot the actual values and the smoothed forecasts on a single graph, we get:



The forecasts from the multiplicative exponential smoothing method do a good job of tracking the seasonal movements in the actual series.

Hodrick-Prescott Filter

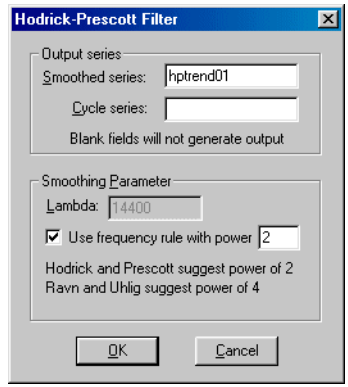
The Hodrick-Prescott Filter is a smoothing method that is widely used among macroeconomists to obtain a smooth estimate of the long-term trend component of a series. The method was first used in a working paper (circulated in the early 1980's and published in 1997) by Hodrick and Prescott to analyze postwar U.S. business cycles.

Technically, the Hodrick-Prescott (HP) filter is a two-sided linear filter that computes the smoothed series s of y by minimizing the variance of y around s, subject to a penalty that constrains the second difference of s. That is, the HP filter chooses s to minimize:

$$\sum_{t=1}^{T}(y_t - s_t)^2 + \lambda \sum_{t=2}^{T-1}((s_{t+1} - s_t) - (s_t - s_{t-1}))^2. \qquad (11.49)$$

The penalty parameter λ controls the smoothness of the series σ. The larger the λ, the smoother the σ. As $\lambda = \infty$, s approaches a linear trend.

To smooth the series using the Hodrick-Prescott filter, choose **Proc/Hodrick-Prescott Filter…**:



First, provide a name for the smoothed series. EViews will suggest a name, but you can always enter a name of your choosing. Next, specify an integer value for the smoothing parameter, λ. You may specify the parameter using the frequency power rule of Ravn and Uhlig (2002) (the number of periods per year divided by 4, raised to a power, and multiplied by 1600), or you may specify λ directly. The default is to use a power rule of 2, yielding the original Hodrick and Prescott values for λ:

$$\lambda = \begin{cases} 100 & \text{for annual data} \\ 1{,}600 & \text{for quarterly data} \\ 14{,}400 & \text{for monthly data} \end{cases} \tag{11.50}$$

Ravan and Uhlig recommend using a power value of 4. EViews will round any non-integer values that you enter. When you click **OK**, EViews displays a graph of the filtered series together with the original series. Note that only data in the current workfile sample are filtered. Data for the smoothed series outside the current sample are filled with NAs.

Frequency (Band-Pass) Filter

EViews computes several forms of band-pass (frequency) filters. These filters are used to isolate the cyclical component of a time series by specifying a range for its duration. Roughly speaking, the band-pass filter is a linear filter that takes a two-sided weighted moving average of the data where cycles in a "band", given by a specified lower and upper bound, are "passed" through, or extracted, and the remaining cycles are "filtered" out.

To employ a band-pass filter, the user must first choose the range of durations (*periodicities*) to pass through. The range is described by a pair of numbers (P_L, P_U), specified in units of the workfile frequency. Suppose, for example, that you believe that the business cycle lasts somewhere from 1.5 to 8 years so that you wish to extract the cycles in this

range. If you are working with quarterly data, this range corresponds to a low duration of 6, and an upper duration of 32 quarters. Thus, you should set $P_L = 6$ and $P_U = 32$.

In some contexts, it will be useful to think in terms of frequencies which describe the number of cycles in a given period (obviously, periodicities and frequencies are inversely related). By convention, we will say that periodicities in the range (P_L, P_U) correspond to frequencies in the range $(2\pi/P_U, 2\pi/P_L)$.

Note that since saying that we have a cycle with a period of 1 is meaningless, we require that $2 \le P_L < P_U$. Setting P_L to the lower-bound value of 2 yields a *high-pass filter* in which all frequencies above $2\pi/P_U$ are passed through.

The various band-pass filters differ in the way that they compute the moving average:

- The fixed length symmetric filters employ a fixed lead/lag length. Here, the user must specify the fixed number of lead and lag terms to be used when computing the weighted moving average. The symmetric filters are time-invariant since the moving average weights depend only on the specified frequency band, and do not use the data. EViews computes two variants of this filter, the first due to Baxter-King (1999) (BK), and the second to Christiano-Fitzgerald (2003) (CF). The two forms differ in the choice of objective function used to select the moving average weights.
- Full sample asymmetric – this is the most general filter, where the weights on the leads and lags are allowed to differ. The asymmetric filter is time-varying with the weights both depending on the data and changing for each observation. EViews computes the Christiano-Fitzgerald (CF) form of this filter.

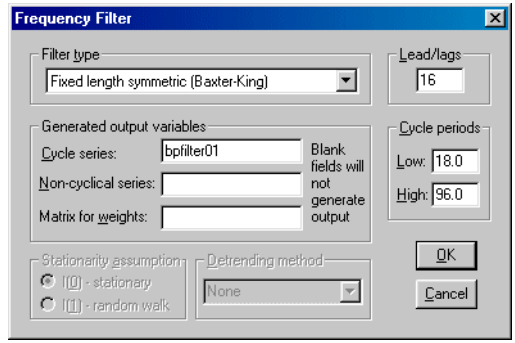
In choosing between the two methods, bear in mind that the fixed length filters require that we use same number of lead and lag terms for every weighted moving average. Thus, a filtered series computed using q leads and lags observations will lose q observations from both the beginning and end of the original sample. In contrast, the asymmetric filtered series do not have this requirement and can be computed to the ends of the original sample.

Computing a Band-Pass Filter in EViews

The band-pass filter is available as a series Proc in EViews. To display the band-pass filter dialog, select **Proc/Frequency Filter...** from the main series menu.

The first thing you will do is to select a filter type. There are three types: **Fixed length symmetric (Baxter-King)**, **Fixed length symmetric (Christiano-Fitzgerald)**, or **Full length asymmetric (Christiano-Fitzgerald)**. By default, the EViews will compute the Baxter-King fixed length symmetric filter.

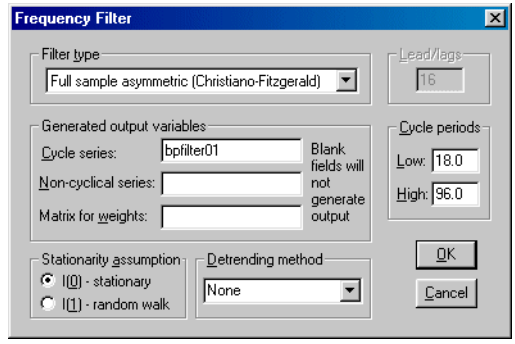
For the Baxter-King filter, there are only a few options that require your attention. First, you must select a frequency length (lead/lags) for the moving average, and the low and high values for the cycle period (P_L, P_U) to be filtered. By default, these fields will be filled in with reasonable default values that are based on the type of your workfile.



Lastly, you may enter the names of objects to contain saved output for the cyclical and non-cyclical components. The **Cycle series** will be a series object containing the filtered series (cyclical component), while the **Non-cyclical series** is simply the difference between the actual and the filtered series. The user may also retrieve the moving average weights used in the filter. These weights, which will be placed in a matrix object, may be used to plot customized frequency response functions. Details are provided below in "The Weight Matrix" on page 361.

Both of the CF filters (symmetric and asymmetric) provide you with additional options for handling trending data.

The first setting involves the **Stationarity assumption**. For both of the CF, you will need to specify whether the series is assumed to be an I(0) covariance stationary process or an I(1) unit root process.

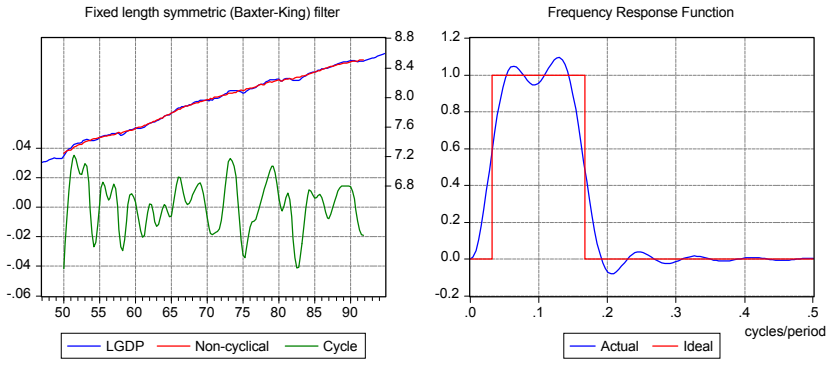


Lastly, you will select a **Detrending method** using the combo. For a covariance stationary series, you may choose to demean or detrend the data prior to applying the filters. Alternatively, for a unit root process, you may choose to demean, detrend, or remove drift using the adjustment suggested by Christiano and Fitzgerald (1999).

Note that, as the name suggests, the full sample filter uses all of the observations in the sample, so that the **Lead/Lags** option is not relevant. Similarly, detrending the data is not an option when using the BK fixed length symmetric filter. The BK filter removes up to two unit roots (a quadratic deterministic trend) in the data so that detrending has no effect on the filtered series.

The Filter Output

Here, we depict the output from the Baxter-King filter. The left panel depicts the original series, filtered series, and the non-cyclical component (difference between the original and the filtered).



For the BK and CF fixed length symmetric filters, EViews plots the frequency response function $\alpha(\omega)$ representing the extent to which the filtered series "responds" to the original series at frequency ω. At a given frequency ω, $|\alpha(\omega)|^2$ indicates the extent to which a moving average raises or lowers the variance of the filtered series relative to that of the original series. The right panel of the graph depicts the function. Note that the horizontal axis of a frequency response function is always in the range 0 to 0.5, in units of cycles per duration. Thus, as depicted in the graph, the frequency response function of the ideal band-pass filter for periodicities (P_L, P_U) will be one in the range $(1/P_U, 1/P_L)$.

The frequency response function is not drawn for the CF time-varying filter since these filters vary with the data and observation number. If you wish to plot the frequency response function for a particular observation, you will have to save the weight matrix and then evaluate the frequency response in a separate step. The example program BFP02.PRG and subroutine FREQRESP.PRG illustrate the steps in computing of gain functions for time-varying filters at particular observations.

The Weight Matrix

For time-invariant (fixed-length symmetric) filters, the weight matrix is of dimension $1 \times (q+1)$ where q is the user-specified lag length order. For these filters, the weights on the leads and the lags are the same, so the returned matrix contains only the one-sided weights. The filtered series can be computed as:

$$z_t = \sum_{c=1}^{q+1} w(1,c) y_{t+1-c} + \sum_{c=2}^{q+1} w(1,c) y_{t+c-1} \qquad t = q+1, \ldots, n-q$$

For time-varying filters, the weight matrix is of dimension $n \times n$ where n is the number of non-missing observations in the current sample. Row r of the matrix contains the weighting vector used to generate the r-th observation of the filtered series where column c contains the weight on the c-th observation of the original series:

$$z_t = \sum_{c=1}^{n} w(t, c) y_c \qquad t = 1, \ldots, n \tag{11.51}$$

where z_t is the filtered series, y_t is the original series and $w(r, c)$ is the (r, c) element of the weighting matrix. By construction, the first and last rows of the weight matrix will be filled with missing values for the symmetric filter.

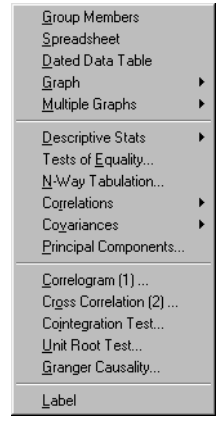
Chapter 12. Groups

This chapter describes the views and procedures of a group object. With a group, you can compute various statistics that describe the relationship between multiple series and display them in various forms such as spreadsheets, tables, and graphs.

The remainder of this chapter assumes that you are already familiar with the basics of creating and working with a group. See the documentation of EViews features beginning with Chapter 4, “Object Basics”, on page 73 for relevant details on the basic operations.

Group Views Overview

The group view menu is divided into four blocks:



- The views in the first block provide various ways of looking at the actual data in the group.
- The views in the second block display various basics statistics.
- The views in the third block are for specialized statistics typically computed using time series data.
- The fourth block contains the label view, which provides information regarding the group object.

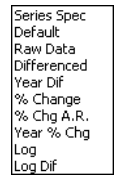
Group Members

This view displays the member series in the group and allows you to alter the group. To change the group, simply edit the group window. You can add other series from the workfile, include expressions involving series, or you can delete series from the group.

Note that editing the window does not change the list of group members. Once you make your changes to the list, you must press the **UpdateGroup** button in the group window toolbar to save the changes.

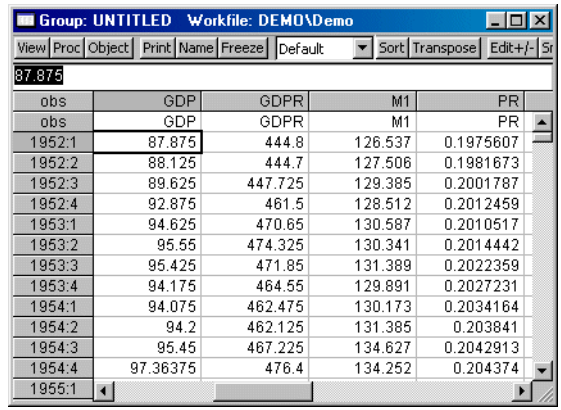
Spreadsheet

This view displays the data, in spreadsheet form, for each series in the group. If you wish, you can flip the rows and columns of the spreadsheet by pressing the **Transpose** button. In transpose format, each row contains a series, and each column an observation or date. Pressing the **Transpose** button toggles between the two spreadsheet views.



You may change the display mode of your spreadsheet view to show various common transformations of your data using the dropdown menu in the group toolbar. By default, EViews displays the original or mapped values in the series using the formatting specified in the series (**Default**). If you wish, you can change the spreadsheet display to show any transformations defined in the individual series (**Series Spec**), the underlying series data (**Raw Data**), or various differences of the series (in levels or percent changes), with or without log transformations.

You may edit the series data in either levels or transformed values. The **Edit +/-** on the group toolbar toggles the edit mode for the group. If you are in edit mode, an edit window appears in the top of the group window and a double-box is used to indicate the cell that is being edited.



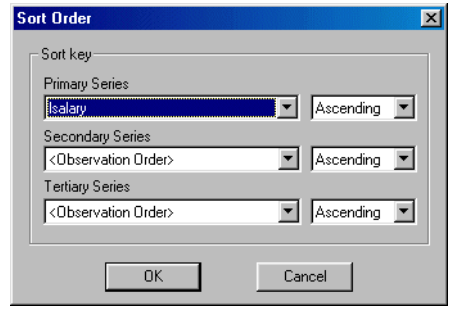
obs	GDP	GDPR	M1	PR
obs	GDP	GDPR	M1	PR
1952:1	87.875	444.8	126.537	0.1975607
1952:2	88.125	444.7	127.506	0.1981673
1952:3	89.625	447.725	129.385	0.2001787
1952:4	92.875	461.5	128.512	0.2012459
1953:1	94.625	470.65	130.587	0.2010517
1953:2	95.55	474.325	130.341	0.2014442
1953:3	95.425	471.85	131.389	0.2022359
1953:4	94.175	464.55	129.891	0.2027231
1954:1	94.075	462.475	130.173	0.2034164
1954:2	94.2	462.125	131.385	0.203841
1954:3	95.45	467.225	134.627	0.2042913
1954:4	97.36375	476.4	134.252	0.204374
1955:1				

Here, we are editing the data in the group in 1-period percent changes (note the label to the right of the edit field). If we change the 1952Q4 value of the percent change in GDP, from 3.626 to 5, the values of GDP from 1952Q4 to the end of the workfile will change to reflect the one-time increase in the value of GDP.

EViews provides you with additional tools for altering the display of your spreadsheet. To change the display properties, select one or more series by clicking on the series names in the headers, then right-click to bring up a menu.

If you right-click and then select **Display format...** EViews will open a format dialog that will allow you to override the individual series display characteristics. Once you specify the desired format and click on **OK**, EViews will update the group display to reflect your specification.

Note that, by default, changes to the group display format will apply only to the group spreadsheet and will not change the underlying series characteristics. If, for example, you elect to show series X in fixed decimal format in the group spreadsheet, but X uses significant digits in its individual series settings, the latter settings will not be modified. To update the display settings in the selected series, you must select the **Apply to underlying series** checkbox in the format dialog.



Most of the other right-mouse menu items should be self-explanatory. Note that you may choose to sort the observations in the group by selecting **Sort...**. EViews will open the Sort Order dialog, prompting you to select sort keys and orders for up to three series.

When you click on **OK**, EViews will rearrange the group spreadsheet display so that observations are displayed in the specified order. Note that the underlying data in the workfile is not sorted, only the display of observations and observation identifiers in the group spreadsheet. This method of changing the spreadsheet display may prove useful if you wish to determine the identities of observations with high or low values for some series in the group.

Lastly, you should note that as with series, you may write the contents of the spreadsheet view to a CSV, tab-delimited ASCII text, RTF, or HTML file by selecting **Save table to disk...** and filling out the resulting dialog.

Dated Data Table

The dated data table view is used to construct tables for reporting and presenting data, forecasts, and simulation results. This view displays the series contained in the group in a variety of formats. You can also use this view to perform common transformations and frequency conversions, and to display data at various frequencies in the same table.

For example, suppose you wish to show your quarterly data for the GDP and PR series, with data for each year, along with an annual average, on a separate line:

	1994				1994
GDP	1698.6	1727.9	1746.7	1774.0	1736.8
PR	1.04	1.05	1.05	1.06	1.05
	1995				1995
GDP	1792.3	1802.4	1825.3	1845.5	1816.4
PR	1.07	1.07	1.08	1.09	1.08
	1996				1996
GDP	1866.9	1902.0	1919.1	1948.2	1909.0
PR	1.09	1.10	1.11	1.11	1.10

The dated data table handles all of the work of setting up this table, and computing the summary values.

Alternatively, you may wish to display annual averages for each year up to the last, followed by the last four quarterly observations in your sample:

	1994	1995	1996	96:1	96:2	96:3	96:4
GDP	1736.8	1816.4	1909.0	1866.9	1902.0	1919.1	1948.2
PR	1.05	1.08	1.10	1.09	1.10	1.11	1.11

Again, the dated data table may be used to perform the required calculations and to set up the table automatically.

The dated data table is capable of creating more complex tables and performing a variety of other calculations. Note, however, that the dated data table view is currently available only for annual, semi-annual, quarterly, or monthly workfiles.

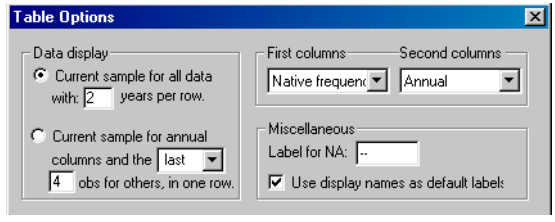
Creating and Specifying a Dated Data Table

To create a dated data table, create a group containing the series of interest and select **View/Dated Data Table**. The group window initially displays a default table view. The default is to show a single year of data on each line, along with a summary measure (the annual average).

You can, however, set options to control the display of your data through the Table and Row Options dialogs. Note the presence of two new buttons on the group window toolbar, labeled **TabOptions** (for Table Options) and **RowOptions. TabOptions** sets the global options for the dated data table. These options will apply to all series in the group object. The **RowOptions** button allows you to override the global options for a particular series. Once you specify table and row options for your group, EViews will remember these options the next time you open the dated data table view for the group.

Table Setup

When you click on the **TabOptions** button, the **Table Option**s dialog appears. The top half of the dialog provides options to control the general style of the table.



The radio buttons on the left hand side of the dialog allow you to choose between the two display formats described above:

- The first style displays the data for n years per row, where n is the positive integer specified in the edit field.
- The second style is a bit more complex. It allows you to specify, for data displayed at a frequency other than annual, the number of observations taken *from the end of the*

workfile sample that are to be displayed. For data displayed at an annual frequency, EViews will display observations over the entire workfile sample.

The two combo boxes on the top right of the dialog supplement your dated display choice by allowing you to display your data at multiple frequencies in each row. The **First Columns** selection describes the display frequency for the first group of columns, while the **Second Columns** selection controls the display for the second group of columns. If you select the same frequency, only one set of results will be displayed.

In each combo box, you may choose among:

- Native frequency (the frequency of the workfile)
- Annual
- Quarterly
- Monthly

If necessary, EViews will perform any frequency conversion (to a lower frequency) required to construct the table.

The effects of these choices on the table display are best described by the following example. For purposes of illustration, note that the current workfile is quarterly, with a current sample of 1993Q1–1996Q4.

Now suppose that you choose to display the first style (two years per row), with the first columns set to the native frequency, and the second columns set to annual frequency. Each row will contain eight quarters of data (the native frequency data) followed by the corresponding two annual observations (the annual frequency data):

	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Year	
	1993				1994				1993	1994
GDP	1611.1	1627.3	1643.6	1676.0	1698.6	1727.9	1746.7	1774.0	1639.5	1736.8
PR	1.02	1.02	1.03	1.04	1.04	1.05	1.05	1.06	1.03	1.05
	1995				1996				1995	1996
GDP	1792.3	1802.4	1825.3	1845.5	1866.9	1902.0	1919.1	1948.2	1816.4	1909.0
PR	1.07	1.07	1.08	1.09	1.09	1.10	1.11	1.11	1.08	1.10

EViews automatically performs the frequency conversion to annual data using the specified method (see "Transformation Methods" on page 368).

If you reverse the ordering of data types in the first and second columns so that the first columns display the annual data, and the second columns display the native frequency, the dated data table will contain:

			Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4
	1993	1994	1993				1994			
GDP	1639.5	1736.8	1611.1	1627.3	1643.6	1676.0	1698.6	1727.9	1746.7	1774.0
PR	1.03	1.05	1.02	1.02	1.03	1.04	1.04	1.05	1.05	1.06
	1995	1996	1995				1996			
GDP	1816.4	1909.0	1792.3	1802.4	1825.3	1845.5	1866.9	1902.0	1919.1	1948.2
PR	1.08	1.10	1.07	1.07	1.08	1.09	1.09	1.10	1.11	1.11

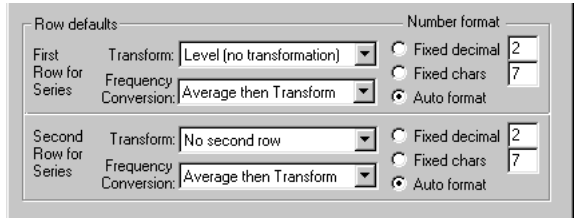
Now, click on **TabOptions**, choose the second display style, and enter `4` in the edit box. Then specify **Annual frequency** for the first columns and **Native frequency** for the second columns. EViews will display the annual data for the current sample, followed by the last four quarterly observations:

	1993	1994	1995	1996	96:1	96:2	96:3	96:4
GDP	1639.5	1736.8	1816.4	1909.0	1866.9	1902.0	1919.1	1948.2
PR	1.03	1.05	1.08	1.10	1.09	1.10	1.11	1.11

Additional Table Options

The bottom of the Table Options dialog controls the default data transformations and numeric display for each series in the group. EViews allows you to use two rows, each with a different transformation and a different output format, to describe each series.

For each row, you specify the transformation method, frequency conversion method, and the number format.



Keep in mind that you may override the default transformation for a particular series using the **RowOptions** menu (p. 371).

Transformation Methods

The following transformations are available:

None (raw data)	No transformation
1 Period Difference	$(y - y(-1))$

1 Year Difference	$y - y(-f)$, where $f = \begin{cases} 1 & \text{for annual} \\ 2 & \text{for semi-annual} \\ 4 & \text{for quarterly} \\ 12 & \text{for monthly} \end{cases}$
1 Period % Change	$100 \times (y - y(-1))/y(-1)$
1 Period % Change at Annual Rate	Computes R such that: $(1 + r/100)^f$ where f is defined above and r is the 1 period % change.
1 Year % Change	$100 \times (y - y(-f))/y(-f)$, where f is defined above.
No second row	Do not display a second row

We emphasize that the above transformation methods represent only the most commonly employed transformations. If you wish to construct your table with other transformations, you should add an appropriate auto-series to the group.

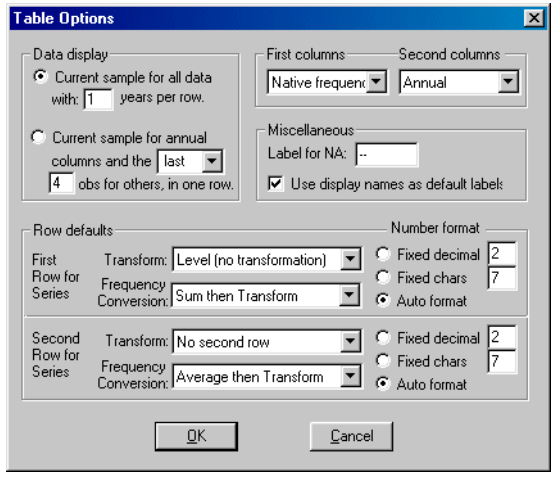
Frequency Conversion

The following frequency conversion methods are provided:

Average then Transform	First convert by taking the average, then transform the average, as specified.
Transform then Average	First transform the series, then take the average of the transformed series.
Sum then Transform	First convert by taking the sum, then transform the sum, as specified.
First Period	Convert by taking the first quarter of each year or first month of each quarter/year.
Last Period	Convert by taking the last quarter of each year or last month of each quarter/year.

The choice between **Average then Transform** and **Transform then Average** changes the ordering of the transformation and frequency conversion operations. The methods differ only for nonlinear transformations (such as the % change methods).

For example, if we specify the dated data table settings:



EViews will display a table with data formatted in the following fashion:

	Q1	Q2	Q3	Q4	Year
			1993		1993
GDP	1611.1	1627.3	1643.6	1676.0	6558.1
PR	1.02	1.02	1.03	1.04	4.11
			1994		1994
GDP	1698.6	1727.9	1746.7	1774.0	6947.1
PR	1.04	1.05	1.05	1.06	4.20
			1995		1995
GDP	1792.3	1802.4	1825.3	1845.5	7265.4
PR	1.07	1.07	1.08	1.09	4.31
			1996		1996
GDP	1866.9	1902.0	1919.1	1948.2	7636.1
PR	1.09	1.10	1.11	1.11	4.41

If, instead, you change the **Frequency Conversion** to **First Period**, EViews will display a table of the form:

	Q1	Q2	Q3	Q4	Year
			1993		1993
GDP	1611.1	1627.3	1643.6	1676.0	1611.1
PR	1.02	1.02	1.03	1.04	1.02
			1994		1994
GDP	1698.6	1727.9	1746.7	1774.0	1698.6
PR	1.04	1.05	1.05	1.06	1.04
			1995		1995
GDP	1792.3	1802.4	1825.3	1845.5	1792.3
PR	1.07	1.07	1.08	1.09	1.07
			1996		1996
GDP	1866.9	1902.0	1919.1	1948.2	1866.9
PR	1.09	1.10	1.11	1.11	1.09

In "Illustration" beginning on page 372, we provide an example which illustrates the computation of the percentage change measures.

Formatting Options

EViews lets you choose between fixed decimal, fixed digit, and auto formatting of the numeric data. Generally, auto formatting will produce appropriate output formatting, but if not, simply select the desired method and enter an integer in the edit field. The options are:

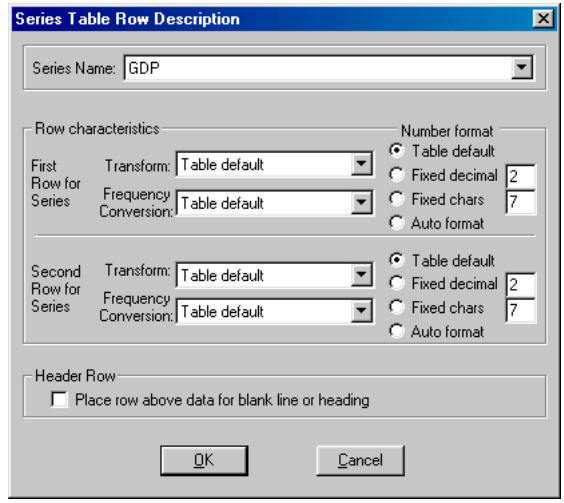
Auto format	EViews chooses the format depending on the data.
Fixed decimal	Specify how many digits to display after the decimal point. This option aligns all numbers at the decimal point.
Fixed chars	Specify how many total characters to display for each number.

EViews will round your data prior to display in order to fit the specified format. This rounding is for display purposes only and does not alter the original data.

Row Options

These options allow you to override the row defaults specified by the **Table Options** dialog. You can specify a different transformation, frequency conversion method, and number format, for each series.

In the **Series Table Row Description** dialog that appears, select the series for which you wish to override the table default options. Then specify the transformation, frequency conversion, or number format you want to use for that series. The options are the same as those described above for the row defaults.



Other Options

Label for NA: allows you to define the symbol used to identify missing values in the table. Bear in mind that if you choose to display your data in transformed form, the transformation may generate missing values even if none of the raw data are missing. Dated data table transformations are explained above.

If your series has display names, you can use the display name as the label for the series in the table by selecting the **Use display names as default labels** option. See Chapter 3 for a discussion of display names and the label view.

Illustration

As an example, consider the following dated data table which displays both quarterly and annual data for GDP and PR in 1995 and 1996:

	1995				1995
GDP	1792.3	1802.4	1825.3	1845.5	1816.4
(% ch.)	1.03	0.56	1.27	1.11	4.58
PR	1.07	1.07	1.08	1.09	1.08
(% ch.)	0.80	0.49	0.52	0.55	2.55

	1996				1996
GDP	1866.9	1902.0	1919.1	1948.2	1909.0
(% ch.)	1.16	1.88	0.90	1.52	5.10
PR	1.09	1.10	1.11	1.11	1.10
(% ch.)	0.72	0.41	0.64	0.46	2.27

At the table level, the first row of output for each of the series is set to be untransformed, while the second row will show the 1-period percentage change in the series. The table defaults have both rows set to perform frequency conversion using the **Average then**

Transformed setting. In addition, we use **Series Table Row Options** dialog to override the second row transformation for PR, setting it to **Transform then Average** option.

The first four columns show the data in native frequency so the choice between **Average then Transform** and **Transform then Average** is irrelevant—each entry in the second row measures the 1-period (1-quarter) percentage change in the variable.

The 1-period percentage change in the last column is computed differently under the two methods. The **Average then Transformed** percentage change in GDP for 1996 measures the percentage change between the average value in 1995 and the average value in 1996. It is computed as:

$$100 \cdot (1909.0 - 1816.3)/1816.4 \cong 5.10 \tag{12.1}$$

EViews computes this transformation using full precision for intermediate results, then displays the result using the specified number format.

The computation of the **Transform then Average** one-period change in PR for 1996 is a bit more subtle. Since we wish to compute measure of the annual change, we first evaluate the one-year percentage change at each of the quarters in the year, and then average the results. For example, the one-year percentage change in 1996Q1 is given by 100(1.09-1.03)/1.03 = 2.29 and the one-year percentage change in 1996Q2 is 100(1.10-1.07)/1.07 = 2.22. Averaging these percentage changes yields:

$$100\left(\frac{1.09 - 1.07}{1.07} + \frac{1.10 - 1.07}{1.07} + \frac{1.11 - 1.08}{1.08} + \frac{1.11 - 1.09}{1.09}\right)/4 \cong 2.27 \tag{12.2}$$

Note also that this computation differs from evaluating the average of the one-quarter percentage changes for each of the quarters of the year.

Other Menu Items

- **Edit +/–** allows you to edit the row (series) labels as well as the actual data in the table. You will not be able to edit any of the computed ranks and any changes that you make to the row labels will only apply to the dated data table view.

  We warn you that *if you edit a data cell, the underlying series data will also change*. This latter feature allows you to use dated data tables for data entry from published sources.

  If you want to edit the data in the table but wish to keep the underlying data unchanged, first **Freeze** the table view and then apply **Edit** to the frozen table.

- **Font** allows you to choose the font, font style, and font size to be used in the table.

- **Title** allows you to add a title to the table.

- **Sample** allows you to change the sample to display in the table.

Here is an example of a table after freezing and editing:

	1995				1995
Gross Domestic Product	1792.3	1802.4	1825.3	1845.5	1816.4
One-period % change	*1.03*	*0.56*	*1.27*	*1.11*	*4.58*
Price Level	1.07	1.07	1.08	1.09	1.08
One-period % change	*0.80*	*0.49*	*0.52*	*0.55*	*2.55*
	1996				1996
Gross Domestic Product	1866.9	1902.0	1919.1	1948.2	1909.0
One-period % change	*1.16*	*1.88*	*0.90*	*1.52*	*5.10*
Price Level	1.09	1.10	1.11	1.11	1.10
One-period % change	*0.72*	*0.41*	*0.64*	*0.46*	*2.27*

Graphs

These views display the series in the group in various graphical forms. You can create graph objects by freezing these views. Chapter 14, “Graphs, Tables, and Text Objects”, on page 415 explains how to edit and modify graph objects in EViews.

The **Graph** views display all series in a single graph. To display each series in a separate graph, see “Multiple Graphs” on page 376.

Line, Area, Bar and Spike Graphs

Displays a line, area, bar or spike graph of the series in the group. Click anywhere in the background of the graph to modify the scaling options or line patterns.

Scatter

There are five variations on the scatter diagram view of a series.

Simple Scatter plots a scatter diagram with the first series on the horizontal axis and the remaining series on the vertical axis.

The remaining three options, **Scatter with Regression**, **Scatter with Nearest Neighbor Fit,** and **Scatter with Kernel Fit**, plot fitted lines of the first series against the second on top of the scatter diagram. They differ in how the fitted lines are calculated. All three graph views are described in detail in “Scatter Diagrams with Fit Lines” beginning on page 400.

XY Pairs produces scatterplots of the first series in the group against the second, the third series against the fourth, and so forth.

XY Line

These views plot XY line graphs of the series in the group. They are similar to the scatterplot graphs, but with successive observations connected by lines. **One X against all Y's** will plot the first series in the group against all other series in the group. **XY pairs** will produce XY plots for successive pairs of series in the group.

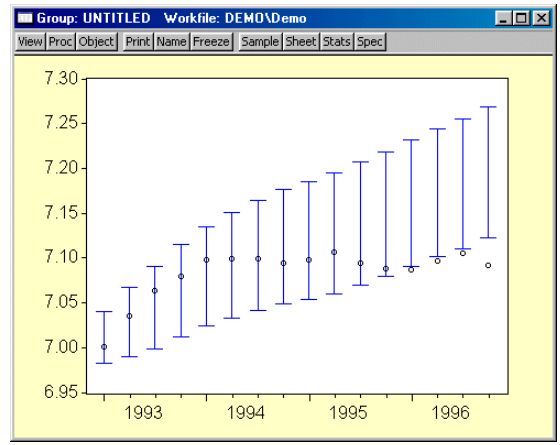
Once you have constructed the XY plot, you may elect to display symbols only (similar to a scatterplot), or lines and symbols for each XY graph. Click anywhere in the background of the view and change the line attributes for the selected line from **Lines only** to **Symbol only** or **Line & Symbol**.

See Chapter 14, "Graphs, Tables, and Text Objects", on page 415 for additional details on graph customization.

Error Bar

The Error Bar view plots error bars using the first two or three series in the group. The first series is used for the "high" value and the second series is the "low" value. The high and low values are connected with a vertical line. The (optional) third series is plotted as a small circle.

Note that EViews does not check the values of your high and low data for consistency. If the high value is *below* the low value, EViews will draw "outside half-lines" that do not connect.



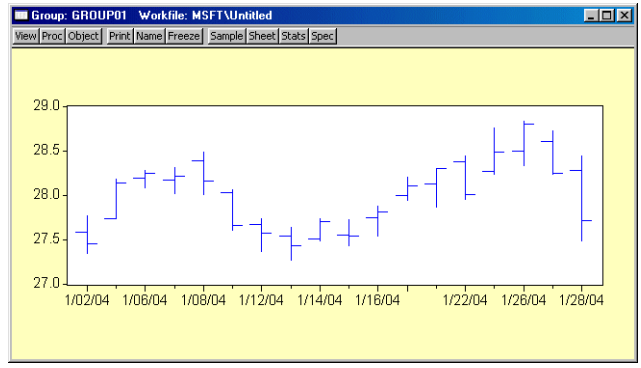
This view is commonly used to display confidence intervals for a statistic.

High-Low (Open-Close)

This view plots the first two to four series in the group as a high-low (open-close) chart. As the name suggests, this chart is commonly used by financial analysts to plot the daily high, low, opening and closing values of stock prices.

The first series in your group should be used to represent the "high" value and the second series should be the "low" value. The high and low values are connected with a vertical line. EViews will not check the values of your high and low data for consistency. If the

high value is *below* the low value, EViews will draw "outside half-lines" that do not connect.



The third and fourth series are optional. If you provide only three series, the third series will be used as the "close" value in a high-low-close graph. The third series will be plotted as a right-facing horizontal line representing the "close" value.

If you provide four series, the third series will represent the "open" value and will be plotted as a left-facing horizontal line. The fourth series will be used to represent the "close" value. The close value will be plotted as a right-facing horizontal line.

Pie

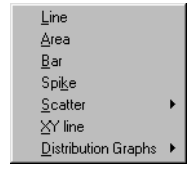
This view displays each observation as a pie chart, where the *percentage* of each series in the group is shown as a wedge in the pie.

If a series has a negative or missing value, the series will simply be dropped from the pie for that observation. You can label the observation number to each pie; double click in the background of the pie chart and mark the **Label Pie** option in the **Graph Options** dialog.

Multiple Graphs

While **Graph** views display all series in a single graph, **Multiple Graphs** views display a separate graph for each series in the group.

Line, Area, Bar and Spike Graphs



These views display a separate line, area, or bar graph for each series in the group.

Scatter

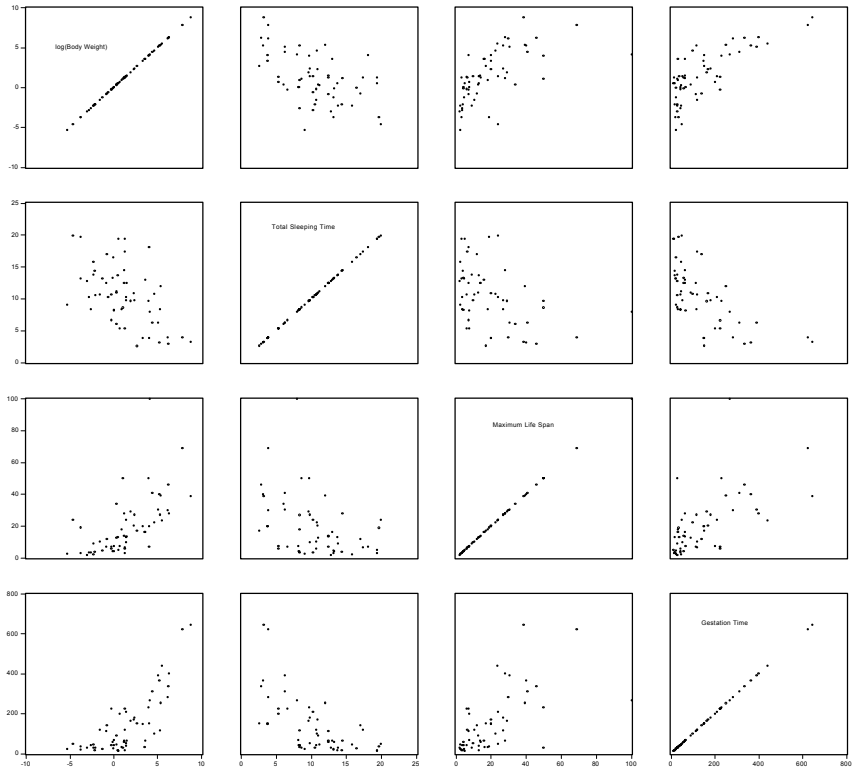
First series against all

This view displays scatter plots with the first series in the group on the horizontal axis and the remaining series on the vertical axis, each in a separate graph. If there are G series in the group, $G-1$ scatter plots will be displayed.

Matrix of all pairs (SCATMAT)

This view displays the *scatterplot matrix*, where scatter plots for all possible pairs of series in the group are displayed as a matrix. The important feature of the scatterplot matrix is that the scatter plots are arranged in such a way that plots in the same column share a common horizontal scale, while plots in the same row share a common vertical scale.

If there are G series in the group, G^2 scatter plots will be displayed. The G plots on the main diagonal all lie on a 45 degree line, showing the distribution of the corresponding series on the 45 degree line. The $G(G-1)/2$ scatters below and above the main diagonal are the same; they are repeated so that we can scan the plots both across the rows and across the columns.



Here is a scatter plot matrix that we copy-and-pasted directly into our document. Note that the resolution of the scatter plot matrix deteriorates quickly as the number of series in the group increases. You may want to freeze the view and modify the graph by moving the axis labels into the scatters on the main diagonal. You can also save more space by moving

each scatter close to each other. Set the vertical and horizontal spacing by right-clicking and choosing the **Position and align graphs...** option.

XY line

This view plots the XY line graph of the first series on the horizontal X-axis and each of the remaining series on the vertical Y-axis in separate graphs. See the XY line view for **Graph** for more information on XY line graphs. If there are G series in the group, $G-1$ XY line graphs will be displayed.

Distribution Graphs

CDF-Survivor-Quantile

This view displays the empirical cumulative distribution functions (CDF), survivor functions, and quantiles of each series in the group. These are identical to the series **CDF-Survivor-Quantile** view; see "CDF-Survivor-Quantile" on page 391 for a detailed description of how these graphs are computed and the available options.

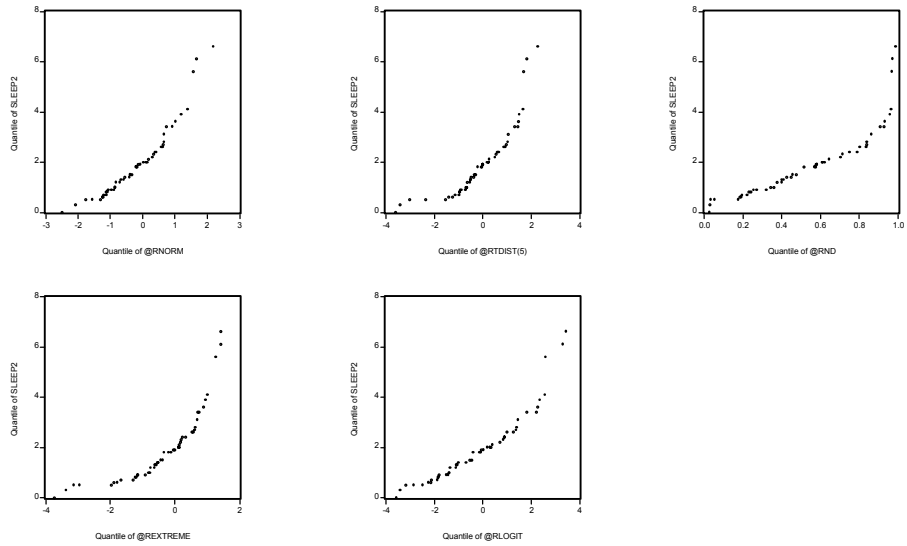
Quantile-Quantile

This view plots the quantiles of each series against the quantiles of a specified distribution or the empirical quantiles of another series. QQ-plots are explained in detail in "Quantile-Quantile" on page 393.

One useful application of group QQ-plots is to form a group of simulated series from different distributions and plot them against the quantiles of the series of interest. This way you can view, at a glance, the QQ-plots against various distributions. Suppose you want to know the distribution of the series SLEEP2. First, create a group containing random draws from the candidate distributions. For example,

```
group dist @rnorm @rtdist(5) @rextreme @rlogit @rnd
```

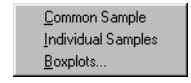
creates a group named DIST that contains simulated random draws from the standard normal, a t-distribution with 5 degrees of freedom, extreme value, logistic, and uniform distributions. Open the group DIST, choose **View/Multiple Graphs/Distribution Graphs/Quantile-Quantile**, select the **Series or Group** option and type in the name of the series SLEEP2 in the field of the QQ Plot dialog box.



The quantiles of SLEEP2 are plotted on the vertical axis of each graph. (We moved one of the graphs to make the plots a bit easier to see.) The QQ-plot of the underlying distribution should lie on a straight line. In this example, none of the QQ-plots lie on a line, indicating that the distribution of SLEEP2 does not match any of those in the group DIST.

Descriptive Statistics

The first two views display the summary statistics of each series in the group. Details for each statistic are provided in "Descriptive Statistics" on page 310.



- **Common Sample** computes the statistics using observations for which there are no missing values in any of the series in the group (casewise deletion of observations).
- **Individual Samples** computes the statistics using all nonmissing observations for each series (listwise deletion).

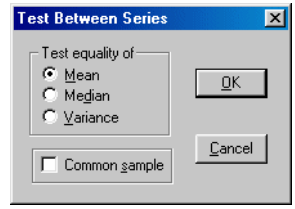
The two views are identical if there are no missing values, or if every series has missing observations at the same observation numbers.

In addition, you may elect to display a statistical graph containing boxplots:

- **Boxplots** computes and displays boxplots for each series. See "Boxplots" on page 409 for details.

Tests of Equality

This view tests the null hypothesis that all series in the group have the same mean, median (distribution), or variance. All of these tests are described in detail in "Equality Tests by Classification" on page 318.



The **Common sample** option uses only observations for which none of the series in the group has missing values.

As an illustration, we demonstrate the use of this view to test for groupwise heteroskedasticity. Suppose we use data for seven countries over the period 1950–1992 and estimate a pooled OLS model (see Chapter 27, "Pooled Time Series, Cross-Section Data", on page 825). To test whether the residuals from this pooled regression are groupwise heteroskedastic, we test the equality of the variances of the residuals for each country.

First, save the residuals from the pooled OLS regression and make a group of the residuals corresponding to each country. This is most easily done by estimating the pooled OLS regression using a pool object and saving the residuals by selecting **Proc/Make Residuals** in the pool object menu or toolbar.

Next, open a group containing the residual series. One method is to highlight each residual series with the right mouse button, double click in the highlighted area and select **Open Group**. Alternatively, you can type `show`, followed by the names of the residual series, in the command window.

Select **View/Tests of Equality...**, and choose the **Variance** option in the **Test Between Series** dialog box.

Test for Equality of Variances between Series
Date: 10/20/97 Time: 15:24
Sample: 1950 1992
Included observations: 43

Method	df	Value	Probability
Bartlett	6	47.65089	1.39E-08
Levene	(6, 287)	5.947002	7.15E-06
Brown-Forsythe	(6, 287)	4.603232	0.000176

Category Statistics

Variable	Count	Std. Dev.	Mean Abs. Mean Diff.	Mean Abs. Median Diff.
RESID_CAN	42	387.3328	288.2434	275.5092
RESID_FRA	42	182.4492	143.0463	140.4258
RESID_GER	42	224.5817	169.6377	167.0994
RESID_ITA	42	173.4625	132.1824	131.2676
RESID_JAP	42	230.4443	185.5166	185.5166
RESID_UK	42	218.8625	159.4564	157.8945
RESID_US	42	340.9424	271.5252	265.4067
All	294	263.4411	192.8011	189.0171

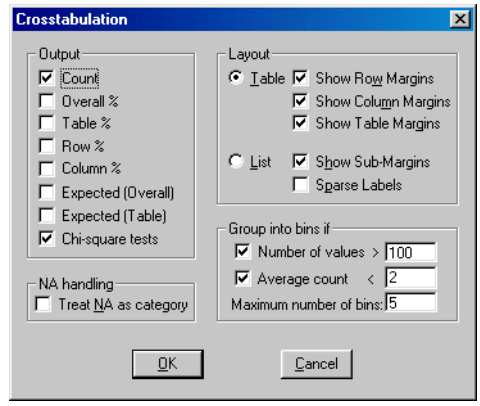
Bartlett weighted standard deviation: 262.1580

The test statistics decisively reject the null hypothesis of equal variance of the residuals across countries, providing strong evidence of the presence of groupwise heteroskedasticity. You may want to adjust the denominator degrees of freedom to take account of the number of estimated parameters in the regression. The tests are, however, consistent even without the degrees of freedom adjustment.

N-Way Tabulation

This view classifies the observations in the current sample into cells defined by the series in the group. You can display the cell counts in various forms and examine statistics for independence among the series in the group. **Select View/N-Way Tabulation...** which opens the tabulation dialog.

Many of the settings will be familiar from our discussion of one-way tabulation in "One-Way Tabulation" on page 325.



Group into Bins If

If one or more of the series in the group is continuous and takes many distinct values, the number of cells becomes excessively large. This option provides you two ways to automatically bin the values of the series into subgroups.

- **Number of values** option bins the series if the series takes more than the specified number of distinct values.
- **Average count** option bins the series if the average count for each distinct value of the series is less than the specified number.
- **Maximum number of bins** specifies the approximate maximum number of subgroups to bin the series. The number of bins may be chosen to be smaller than this number in order to make the bins approximately the same size.

The default setting is to bin a series into approximately 5 subgroups if the series takes more than 100 distinct values or if the average count is less than 2. If you do not want to bin the series, unmark both options.

NA Handling

By default, EViews drops observations from the contingency table where any of the series in the group has a missing value. **Treat NA as category** option includes all observations and counts NAs in the contingency table as an explicit category.

Layout

This option controls the display style of the tabulation. The **Table** mode displays the categories of the first two series in $r \times c$ tables for each category of the remaining series in the group.

The **List** mode displays the table in a more compact, hierarchical form. The **Sparse Labels** option omits repeated category labels to make the list less cluttered. Note that some of the conditional χ^2 statistics are not displayed in list mode.

Output

To understand the options for output, consider a group with three series. Let (i, j, k) index the bin of the first, second, and third series, respectively. The number of observations in the (i, j, k)-th cell is denoted as n_{ijk} with a total of $N = \sum_i \sum_j \sum_k n_{ijk}$ observations.

- **Overall%** is the percentage of the total number of observations accounted for by the cell count.
- **Table%** is the percentage of the total number of observations in the conditional table accounted for by the cell count.
- **Row%** is the percentage of the number of observations in the row accounted for by the cell count.
- **Column%** is the percentage of the number of observations in the column accounted for by the cell count.

The *overall* expected count in the (i, j, k)-th cell is the number expected if *all* series in the group were independent of each other. This expectation is estimated by:

$$\hat{n}_{ijk} = (\sum_i n_{ijk^*}/N)(\sum_j n_{ijk^*}/N)(\sum_k n_{ijk^*}/N)N. \tag{12.3}$$

The *table* expected count $\tilde{n}_{ijk}$ is estimated by computing the expected count for the conditional table. For a given table, this expected value is estimated by:

$$\tilde{n}_{ijk^*} = (\sum_i n_{ijk^*}/N_{k^*})(\sum_j n_{ijk^*}/N_{k^*})N_{k^*} \tag{12.4}$$

where N_{k^*} is the total number of observations in the k^* table.

Chi-square Tests

If you select the **Chi-square tests** option, EViews reports χ^2 statistics for testing the independence of the series in the group. The test statistics are based on the distance between the actual cell count and the count expected under independence.

- **Overall (unconditional) independence among all series in the group**. EViews reports the following two test statistics for overall independence among all series in the group:

$$\begin{aligned} \text{Pearson } \chi^2 &= \sum_{i,j,k} \frac{(\hat{n}_{i,j,k} - n_{i,j,k})^2}{\hat{n}_{i,j,k}} \\ \text{Likelihood ratio} &= 2 \sum_{i,j,k} n_{i,j,k} \log\left(\frac{n_{i,j,k}}{\hat{n}_{i,j,k}}\right) \end{aligned} \tag{12.5}$$

  where n_{ijk} and $\hat{n}_{ijk}$ are the actual and overall expected count in each cell. Under the null hypothesis of independence, the two statistics are asymptotically distributed

χ^2 with $IJK-(I-1)-(J-1)-(K-1)-1$ degrees of freedom where I, J, K are the number of categories for each series.

These test statistics are reported at the top of the contingency table:

Tabulation of LWAGE and UNION and MARRIED
Date: 012/15/00 Time: 14:12
Sample: 1 1000
Included observations: 1000

Tabulation Summary

Variable	Categories		
LWAGE	5		
UNION	2		
MARRIED	2		
Product of Categories	20		
Test Statistics	**df**	**Value**	**Prob**
Pearson X2	13	174.5895	0.0000
Likelihood Ratio G2	13	167.4912	0.0000

WARNING: Expected value is less than 5 in 40.00% of cells (8 of 20).

In this group, there are three series LWAGE, UNION, and MARRIED, each with $I = 5$, $J = 2$, and $K = 2$ categories. Note the WARNING message: if there are many cells with expected value less than 5, the small sample distribution of the test statistic under the null hypothesis may deviate considerably from the asymptotic χ^2 distribution.

- **Conditional independence between series in the group**. If you display in table mode, EViews presents measures of association for *each* conditional table. These measures are analogous to the correlation coefficient; the larger the measure, the larger the association between the row series and the column series in the table. In addition to the Pearson χ^2 for the table, the following three measures of association are reported:

$$\text{Phi coefficient} = \sqrt{\tilde{\chi}^2/\tilde{N}} \tag{12.6}$$

$$\text{Cramers V} = \sqrt{\tilde{\chi}^2/((\min\{r,c\}-1)\tilde{N})} \tag{12.7}$$

$$\text{Contingency coefficient} = \sqrt{\tilde{\chi}^2/(\tilde{\chi}^2+N)} \tag{12.8}$$

where $\min(r, c)$ is the smaller of the number of row categories r or column categories c of the table, and $\tilde{N}$ is the number of observations in the table. Note that all three measures are bounded between 0 and 1, a higher number indicating a stronger relation between the two series in the table. While the correlation coefficient only measures the linear association between two series, these nonparametric measures are robust to departures from linearity.

Table 1: Conditional table for MARRIED=0:

Count		UNION 0	1	Total
LWAGE	[0, 1)	0	0	0
	[1, 2)	167	8	175
	[2, 3)	121	44	165
	[3, 4)	17	2	19
	[4, 5)	0	0	0
	Total	305	54	359

Measures of Association	Value
Phi Coefficient	0.302101
Cramer's V	0.302101
Contingency Coefficient	0.289193

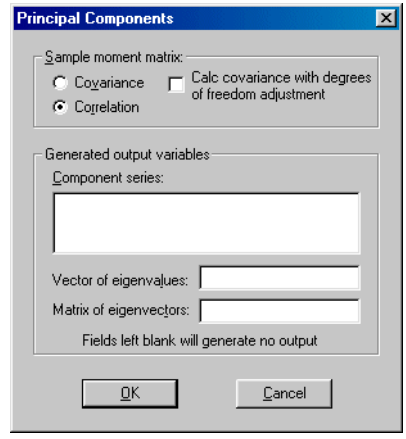
Table Statistics	df	Value	Prob
Pearson X2	2	32.76419	7.68E-08
Likelihood Ratio G2	2	34.87208	2.68E-08

Note: Expected value is less than 5 in 16.67% of cells (1 of 6).

Bear in mind that these measures of association are computed for each two-way table. The conditional tables are presented at the top, and the unconditional tables are reported at the bottom of the view.

Principal Components

The principal components view of the group displays the Eugene-decomposition of the sample second moment of a group of series. Select **View/Principal Components...** to call up the dialog.



You may either decompose the sample covariance matrix or the correlation matrix computed for the series in the group. The sample second moment matrix is computed using data in the current workfile sample. If there are any missing values, the sample second moment is computed using the *common* sample where observations within the workfile range with missing values are dropped.

There is also a checkbox that allows you to correct for degrees of freedom in the computation of the covariances. If you select this option, EViews will divide the sum of squared deviations by $n - 1$ instead of n.

You may store the results in your workfile by simply providing the names in the appropriate fields. To store the first k principal component series, simply list k names in the Com-

ponent series edit field, each separated by a space. Note that you cannot store more principal components than there are series in the group. You may also store the eigenvalues and eigenvectors in a named vector and matrix.

The principal component view displays output that looks as follows:

Date: 10/31/00 Time: 16:05
Sample: 1 74
Included observations: 74
Correlation of X1 X2 X3 X4

	Comp 1	Comp 2	Comp 3	Comp 4
Eigenvalue	3.497500	0.307081	0.152556	0.042863
Variance Prop.	0.874375	0.076770	0.038139	0.010716
Cumulative Prop.	0.874375	0.951145	0.989284	1.000000

Eigenvectors:

Variable	Vector 1	Vector 2	Vector 3	Vector 4
X1	-0.522714	-0.164109	-0.236056	0.802568
X2	-0.512619	-0.074307	-0.660639	-0.543375
X3	-0.491857	-0.537452	0.640927	-0.241731
X4	0.471242	-0.823827	-0.311521	0.046838

The column headed by "Comp1" and "Vector1" corresponds to the first principal component, "Comp2" and "Vector2" denote the second principal component and so on. The row labeled "Eigenvalue" reports the eigenvalues of the sample second moment matrix in descending order from left to right. The Variance Prop. row displays the variance proportion explained by each principal component. This value is simply the ratio of each eigenvalue to the sum of all eigenvalues. The Cumulative Prop. row displays the cumulative sum of the Variance Prop. row from left to right and is the variance proportion explained by principal components up to that order.

The second part of the output table displays the eigenvectors corresponding to each eigenvalue. The first principal component is computed as a linear combination of the series in the group with weights given by the first eigenvector. The second principal component is the linear combination with weights given by the second eigenvector and so on.

Correlations, Covariances, and Correlograms

Correlations and **Covariances** display the correlation and covariance matrices of the series in the group. The **Common Sample** view drops observations for which any one of the series has missing data in the current sample. The **Pairwise Samples** view computes each of the second moments using all non-missing observations for the relevant series. Note that **Pairwise Samples** returns entries that correspond to `@cov(x,y)` and `@cor(x,y)`

functions. For unbalanced samples, the **Pairwise Samples** method uses the maximum number of observations, but may result in a non-positive definite matrix.

Correlogram displays the autocorrelations and partial autocorrelations of the first series in the group. See "Correlogram" on page 326, for a description of the correlogram view.

Cross Correlations and Correlograms

This view displays the cross correlations of the first two series in the group. The cross correlations between the two series x and y are given by,

$$r_{xy}(l) = \frac{c_{xy}(l)}{\sqrt{c_{xx}(0)} \cdot \sqrt{c_{yy}(0)}}, \quad \text{where } l = 0, \pm 1, \pm 2, \ldots \tag{12.9}$$

and,

$$c_{xy}(l) = \begin{cases} \sum_{t=1}^{T-l} ((x_t - \bar{x})(y_{t+l} - \bar{y}))/T & l = 0, 1, 2, \ldots \\ \sum_{t=1}^{T+l} ((y_t - \bar{y})(x_{t-l} - \bar{x}))/T & l = 0, -1, -2, \ldots \end{cases} \tag{12.10}$$

Note that, unlike the autocorrelations, cross correlations are not necessarily symmetric around lag 0.

The dotted lines in the cross correlograms are the approximate two standard error bounds computed as $\pm 2/(\sqrt{T})$.

Cointegration Test

This view carries out the Johansen test for whether the series in the group are cointegrated or not. "Cointegration Test" on page 739 discusses the Johansen test in detail and describes how one should interpret the test results.

Unit Root Test

This view carries out the Augmented Dickey-Fuller (ADF), GLS transformed Dickey-Fuller (DFGLS), Phillips-Perron (PP), Kwiatkowski, *et. al.* (KPSS), Elliot, Richardson and Stock (ERS) Point Optimal, and Ng and Perron (NP) unit root tests for whether the series in the group (or the first or second differences of the series) are stationary.

See "Panel Unit Root Tests" on page 530 for additional discussion.

Granger Causality

Correlation does not necessarily imply causation in any meaningful sense of that word. The econometric graveyard is full of magnificent correlations, which are simply spurious or meaningless. Interesting examples include a positive correlation between teachers' salaries and the consumption of alcohol and a superb positive correlation between the death rate in the UK and the proportion of marriages solemnized in the Church of England. Economists debate correlations which are less obviously meaningless.

The Granger (1969) approach to the question of whether x causes y is to see how much of the current y can be explained by past values of y and then to see whether adding lagged values of x can improve the explanation. y is said to be Granger-caused by x if x helps in the prediction of y, or equivalently if the coefficients on the lagged x's are statistically significant. Note that two-way causation is frequently the case; x Granger causes y and y Granger causes x.

It is important to note that the statement "x Granger causes y" does not imply that y is the effect or the result of x. Granger causality measures precedence and information content but does not by itself indicate causality in the more common use of the term.

When you select the **Granger Causality** view, you will first see a dialog box asking for the number of lags to use in the test regressions. In general, it is better to use more rather than fewer lags, since the theory is couched in terms of the relevance of all past information. You should pick a lag length, l, that corresponds to reasonable beliefs about the longest time over which one of the variables could help predict the other.

EViews runs bivariate regressions of the form:

$$\begin{aligned} y_t &= \alpha_0 + \alpha_1 y_{t-1} + \ldots + \alpha_l y_{t-l} + \beta_1 x_{t-1} + \ldots + \beta_l x_{-l} + \epsilon_t \\ x_t &= \alpha_0 + \alpha_1 x_{t-1} + \ldots + \alpha_l x_{t-l} + \beta_1 y_{t-1} + \ldots + \beta_l y_{-l} + u_t \end{aligned} \tag{12.11}$$

for all possible pairs of (x, y) series in the group. The reported F-statistics are the Wald statistics for the joint hypothesis:

$$\beta_1 = \beta_2 = \ldots = \beta_l = 0 \tag{12.12}$$

for each equation. The null hypothesis is that x does *not* Granger-cause y in the first regression and that y does *not* Granger-cause x in the second regression. The test results are given by:

Pairwise Granger Causality Tests
Date: 10/20/97 Time: 15:31
Sample: 1946:1 1995:4
Lags: 4

Null Hypothesis:	Obs	F-Statistic	Probability
GDP does not Granger Cause CS	189	1.39156	0.23866
CS does not Granger Cause GDP		7.11192	2.4E-05

For this example, we cannot reject the hypothesis that GDP does not Granger cause CS but we do reject the hypothesis that CS does not Granger cause GDP. Therefore it appears that Granger causality runs one-way from CS to GDP and not the other way.

If you want to run Granger causality tests with other exogenous variables (*e.g.* seasonal dummy variables or linear trends) or if you want to carry out likelihood ratio (LR) tests, run the test regressions directly using equation objects.

Label

This view displays the label information of the group. You can edit any of the field cells in the label view, except the Last Update cell which shows the date/time the group was last modified.

Name is the group name as it appears in the workfile; you can rename your group by editing this cell. If you fill in the Display Name cell, EViews will use this name in some of the tables and graphs of the group view. Unlike Names, Display Names may contain spaces and preserve capitalization (upper and lower case letters).

See Chapter 10, “EViews Databases”, on page 261 for a discussion of the label fields and their use in database searches.

Group Procedures Overview

There are three procedures available for groups.

- **Make Equation…** opens an **Equation Specification** dialog box with the first series in the group listed as the dependent variable and the remaining series as the regressors, including a constant term C. You can modify the specification as desired.
- **Make Vector Autoregression…** opens an **Unrestricted Vector Autoregression** dialog box, where all series in the group are listed as endogenous variables in the VAR. See Chapter 24, “Vector Autoregression and Error Correction Models”, on page 721 for a discussion of specifying and estimating VARs in EViews.
- **Resample...** performs resampling on all of the series in the group. A description of the resampling procedure is provided in “Resample” on page 334.

Chapter 13. Statistical Graphs from Series and Groups

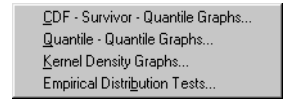
EViews provides several methods for exploratory data analysis. In Chapter 11, "Series", on page 309 we document several graph views that may be used to characterize the distribution of a series. This chapter describes bivariate scatterplot views which allow you to fit lines using parametric, and nonparametric procedures, and boxplot views which may be used to characterize the distribution of your data.

These views, some of which involve relatively complicated calculations or have a number of specialized options, are documented in detail below. While the discussion may sometimes involves fairly technical issues, you should not feel as though you need to master all of the details to use these views. The graphs correspond to familiar concepts, and are designed to be simple and easy to understand visual displays of your data. The EViews default settings should be sufficient for all but the most specialized of analyses. Feel free to explore each of the views, clicking on **OK** to accept the default settings.

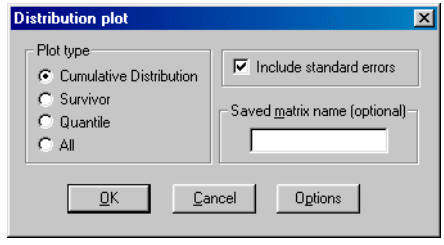
Distribution Graphs of Series

The view menu of a series lists three graphs that characterize the empirical distribution of the series under **View/Distribution...**

CDF-Survivor-Quantile



This view plots the empirical cumulative distribution, survivor, and quantile functions of the series together with the plus or minus two standard error bands. Select **View/Distribution Graphs/CDF-Survivor-Quantile...**



- The **Cumulative Distribution** option plots the empirical cumulative distribution function (CDF) of the series. The CDF is the probability of observing a value from the series not exceeding a specified value r:

$$F_x(r) = \Pr(x \le r). \tag{13.1}$$

- The **Survivor** option plots the empirical survivor function of the series. The survivor function gives the probability of observing a value from the series at least as large as some specified value r and is equal to one minus the CDF:

$$S_x(r) = \Pr(x > r) = 1 - F_x(r) \tag{13.2}$$

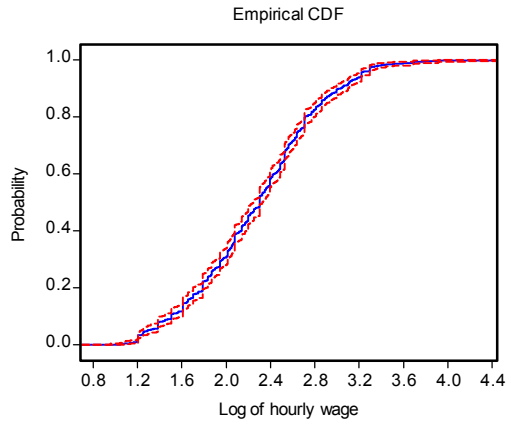
- The **Quantile** option plots the empirical quantiles of the series. For $0 < q < 1$, the q-th quantile $x_{(q)}$ of x is a number such that:

$$\begin{aligned} \Pr(x \le x_{(q)}) &\le q \\ \Pr(x \ge x_{(q)}) &\le 1 - q \end{aligned} \tag{13.3}$$

  The quantile is the inverse function of the CDF; graphically, the quantile can be obtained by flipping the horizontal and vertical axis of the CDF.

- The **All** option plots the CDF, survivor, and quantiles.

For example, working with the series LWAGE containing log wage data, and selecting a CDF plot yields:



Standard Errors

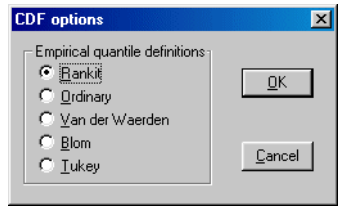
The **Include standard errors** option plots the approximate 95% confidence intervals together with the empirical distribution functions. The methodology for computing these intervals is described in detail in Conover (1980, pp. 114–116). Note that using this approach, we do not compute confidence intervals for the quantiles corresponding to the first and last few order statistics.

Saved matrix name

This optional edit field allows you to save the results in a matrix object. See `cdfplot` (p. 235) of the *Command and Programming Reference* for details on the structure of the saved matrix.

Options

EViews provides several methods of computing the empirical CDF used in the CDF and quantile computations:



Given a total of N observations, the CDF for value r is estimated as:

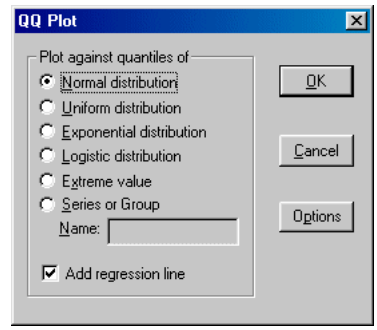
Rankit (default)	$(r-1/2)/N$
Ordinary	r/N
Van der Waerden	$r/(N+1)$
Blom	$(r-3/8)/(N+1/4)$
Tukey	$(r-1/3)/(N+1/3)$

The various methods differ in how they adjust for non-continuity in the CDF computation. The differences between these alternatives will become negligible as the sample size N grows.

Quantile-Quantile

The quantile-quantile (QQ)-plot is a simple yet powerful tool for comparing two distributions (Cleveland, 1994). This view plots the quantiles of the chosen series against the quantiles of another series or a theoretical distribution. If the two distributions are the same, the QQ-plot should lie on a straight line. If the QQ-plot does not lie on a straight line, the two distributions differ along some dimension. The pattern of deviation from linearity provides an indication of the nature of the mismatch.

To generate a QQ-plot, select **View/Distribution Graphs/Quantile-Quantile...** You can plot against the quantiles of the following theoretical distributions:



- **Normal**. Bell-shaped and symmetric distribution.
- **Uniform.** Rectangular density function. Equal probabilities associated with any fixed interval size in the support.
- **Exponential**. The unit exponential is a positively skewed distribution with a long right tail.
- **Logistic**. This symmetric distribution is similar to the normal, except that it has longer tails than the normal.
- **Extreme value**. The Type-I (minimum) extreme value is a negatively skewed distribution with a long left tail—it is very close to a lognormal distribution.

You can also plot against the quantiles of any series in your workfile. Type the names of the series or groups in the edit box, and select **Series** or **Group**. EViews will compute a QQ-plot against each series in the list. You can use this option to plot against the quantiles of a simulated series from any distribution; see the example below.

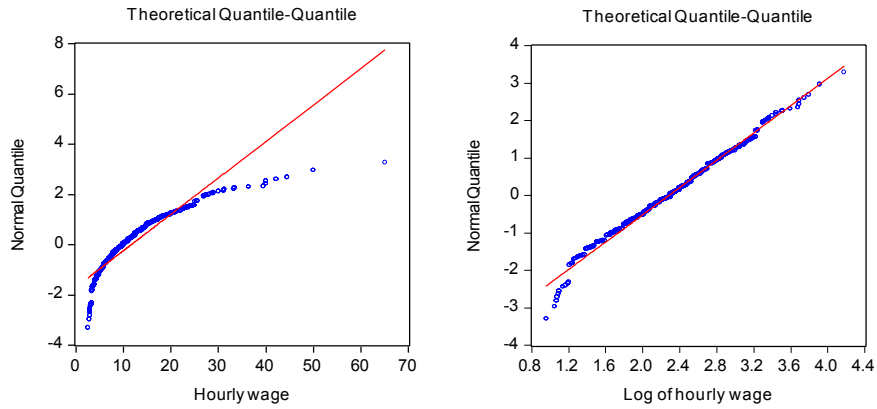
The checkbox provides you with the option of plotting a regression line through the quantile values.

The **Options** button provides you with several methods for computing the empirical quantiles. The options are explained in the CDF-Survivor-Quantile section above; the choice should not make much difference unless the sample is very small.

For additional details, see Cleveland (1994), or Chambers, et al. (1983, Chapter 6).

Illustration

Labor economists typically estimate wage earnings equations with the log of wage on the left-hand side instead of the wage itself. This is because the log of wage has a distribution more close to the normal than the wage, and classical small sample inference procedures are more likely to be valid. To check this claim, we can plot the quantiles of the wage and log of wage against those from the normal distribution. Highlight the series, double click, select **View/Distribution Graphs/Quantile-Quantile...**, and choose the (default) **Normal distribution** option:



If the distributions of the series on the vertical and horizontal axes match, the plots should lie on a straight line. The two plots clearly indicate that the log of wage has a distribution closer to the normal than the wage.

The concave shape of the QQ-plot for the wage indicates that the distribution of the wage series is positively skewed with a long right tail. If the shape were convex, it would indicate that the distribution is negatively skewed.

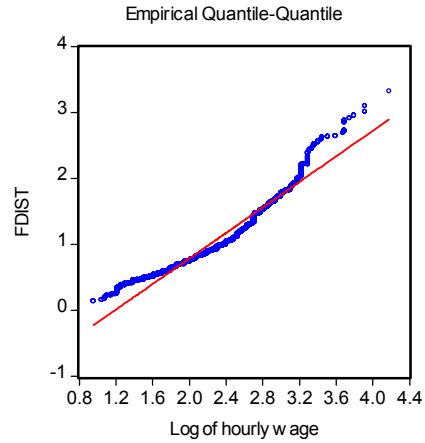
The QQ-plot for the log of wage falls nearly on a straight line except at the left end, where the plot curves downward. QQ-plots that fall on a straight line in the middle but curve upward at the left end and curve downward at the right end indicate that the distribution is leptokurtic and has a thicker tail than the normal distribution. If the plot curves downward at the left, and upward at the right, it is an indication that the distribution is platykurtic and has a thinner tail than the normal distribution. Here, it appears that log wages are somewhat platykurtic.

If you want to compare your series with a distribution not in the option list, you can use the random number generator in EViews and plot against the quantiles of the simulated series from the distribution. For example, suppose we wanted to compare the distribution of the log of wage with the F-distribution with 10 numerator degrees of freedom and 50 denominator degrees of freedom. First generate a random draw from an $F(10,50)$ distribution using the command:

```
series fdist=@rfdist(10,50)
```

Then highlight the log of wage series, double click, select **View/Distribution Graphs/ Quantile-Quantile...**, and choose the **Series or Group** option and type in the name of the simulated series (in this case `fdist`).

The plot is slightly convex, indicating that the distribution of the log of wage is slightly negatively skewed compared to the $F(10,50)$.



Kernel Density

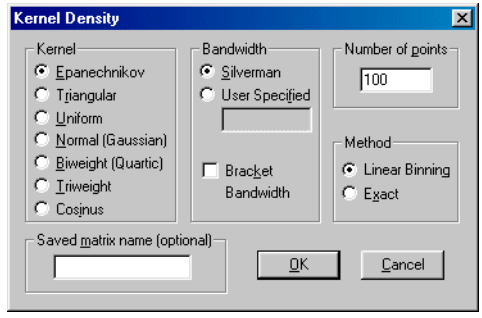
This view plots the kernel density estimate of the distribution of the series. The simplest nonparametric density estimate of a distribution of a series is the histogram. You can view the histogram by selecting **View/Descriptive Statistics/Histogram and Stats**. The histogram, however, is sensitive to the choice of origin and is not continuous.

The kernel density estimator replaces the "boxes" in a histogram by "bumps" that are smooth (Silverman 1986). Smoothing is done by putting less weight on observations that are further from the point being evaluated. More technically, the kernel density estimate of a series X at a point x is estimated by:

$$\mathrm{f}(x) = \frac{1}{Nh} \sum_{i=1}^{N} K\left(\frac{x - X_i}{h}\right), \tag{13.4}$$

where N is the number of observations, h is the bandwidth (or smoothing parameter) and K is a kernel weighting function that integrates to one.

When you choose **View/Distribution Graphs/Kernel Density...**, the **Kernel Density** dialog appears:



To display the kernel density estimates, you need to specify the following:

- **Kernel**. The kernel function is a weighting function that determines the shape of the bumps. EViews provides the following options for the kernel function K:

Epanechnikov (default)	$\frac{3}{4}(1-u^2)I(\lvert u\rvert \le 1)$
Triangular	$(1-\lvert u\rvert)(I(\lvert u\rvert \le 1))$
Uniform (Rectangular)	$\frac{1}{2}(I(\lvert u\rvert \le 1))$
Normal (Gaussian)	$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}u^2\right)$
Biweight (Quartic)	$\frac{15}{16}(1-u^2)^2 I(\lvert u\rvert \le 1)$
Triweight	$\frac{35}{32}(1-u^2)^3 I(\lvert u\rvert \le 1)$
Cosinus	$\frac{\pi}{4}\cos\left(\frac{\pi}{2}u\right)I(\lvert u\rvert \le 1)$

where u is the argument of the kernel function and I is the indicator function that takes a value of one if its argument is true, and zero otherwise.

- **Bandwidth**. The bandwidth h controls the smoothness of the density estimate; the larger the bandwidth, the smoother the estimate. Bandwidth selection is of crucial importance in density estimation (Silverman, 1986), and various methods have been suggested in the literature. The **Silverman** option (default) uses a data-based automatic bandwidth:

$$h = 0.9kN^{-1/5}\min(s, R/1.34) \tag{13.5}$$

where N is the number of observations, s is the standard deviation, and R is the interquartile range of the series (Silverman 1986, equation 3.31). The factor k is a canonical bandwidth-transformation that differs across kernel functions (Marron and Nolan 1989; Härdle 1991). The canonical bandwidth-transformation adjusts the bandwidth so that the automatic density estimates have roughly the same amount of smoothness across various kernel functions.

To specify a bandwidth of your choice, mark **User Specified** option and type a non-negative number for the bandwidth in the field box. Although there is no general rule for the appropriate choice of the bandwidth, Silverman (1986, section 3.4) makes a case for undersmoothing by choosing a somewhat small bandwidth, since it is easier for the eye to smooth than it is to unsmooth.

The **Bracket Bandwidth** option allows you to investigate the sensitivity of your estimates to variations in the bandwidth. If you choose to bracket the bandwidth, EViews plots three density estimates using bandwidths $0.5h$, h, and $1.5h$.

- **Number of Points**. You must specify the number of points M at which you will evaluate the density function. The default is $M = 100$ points. Suppose the minimum and maximum value to be considered are given by X_L and X_U, respectively. Then $f(x)$ is evaluated at M equi-spaced points given by:

$$x_i = X_L + i \cdot \left(\frac{X_U - X_L}{M} \right), \text{ for } i = 0, 1, \ldots M - 1. \tag{13.6}$$

  EViews selects the lower and upper evaluation points by extending the minimum and maximum values of the data by two (for the normal kernel) or one (for all other kernels) bandwidth units.

- **Method**. By default, EViews utilizes the **Linear Binning** approximation algorithm of Fan and Marron (1994) to limit the number of evaluations required in computing the density estimates. For large samples, the computational savings are substantial.

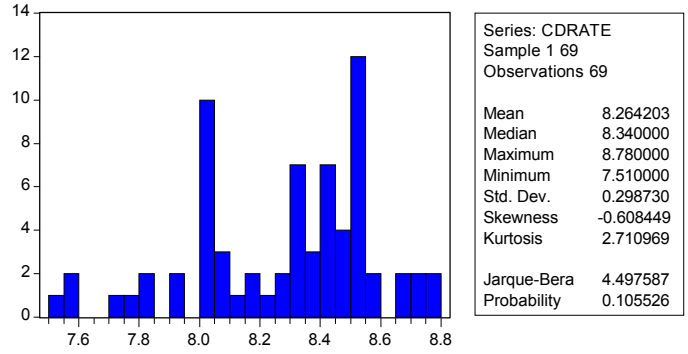
  The **Exact** option evaluates the density function using all of the data points for each X_j, $j = 1, 2, \ldots, N$ for each x_i. The number of kernel evaluations is therefore of order $O(NM)$, which, for large samples, may be quite time-consuming.

  Unless there is a strong reason to compute the exact density estimate or unless your sample is very small, we recommend that you use the binning algorithm.

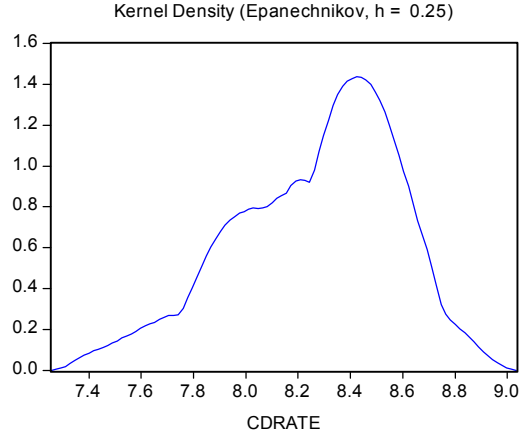
- **Saved matrix name**. This optional edit field allows you to save the results in a matrix object. See `kdensity` (p. 328) in the *Command and Programming Reference* for details on the structure of the saved matrix.

Illustration

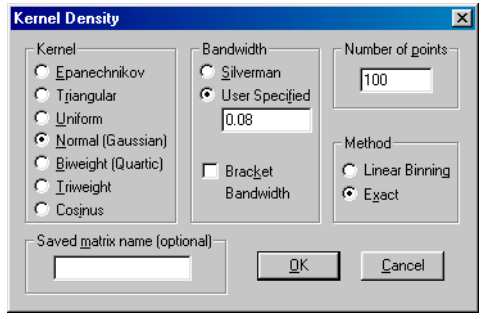
As an illustration of kernel density estimation, we use the three month CD rate data for 69 Long Island banks and thrifts used in Simonoff (1996). The histogram of the CD rate looks as follows:



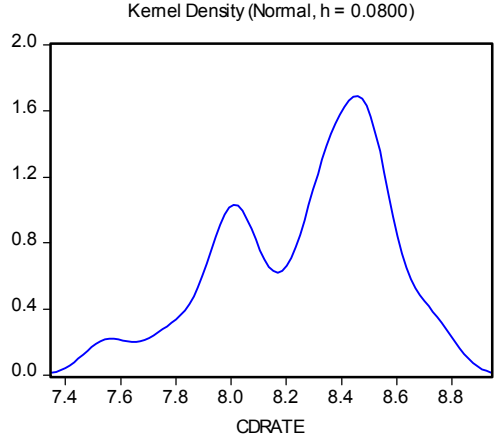
This histogram is a very crude estimate of the distribution of CD rates and does not provide us with much information about the underlying distribution. To view the kernel density estimate, select **View/Distribution Graphs/Kernel Density...** The default options produced the following view:



This density estimate seems to be oversmoothed. Simonoff (1996, chapter 3) uses a Gaussian kernel with bandwidth 0.08. To replicate his results, select **View/Distribution Graphs/Kernel Density...** and fill in the dialog box as follows:



Note that we select the **Exact** method option since there are only 69 observations to evaluate the kernel. The kernel density result is depicted below:



This density estimate has about the right degree of smoothing. Interestingly enough, this density has a trimodal shape with modes at the "focal" numbers 7.5, 8.0, and 8.5.

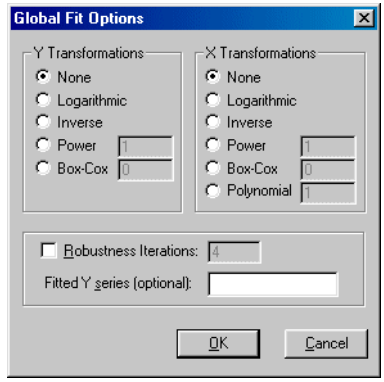
Scatter Diagrams with Fit Lines

The view menu of a group includes four variants of scatterplot diagrams. Click on **View/Graph/Scatter**, then select **Simple Scatter** to plot a scatter diagram with the first series on the horizontal axis and the remaining series on the vertical axis. The **XY Pairs** form of the scatterplot graph, plots scatter diagrams in pairs, with the first series plotted against the second, and the third plotted against the fourth, *etc.*

The remaining three graphs, **Scatter with Regression**, **Scatter with Nearest Neighbor Fit,** and **Scatter with Kernel Fit** plot fitted lines for the scatterplot of the first series against the second series.

Scatter with Regression

This view fits a bivariate regression of transformations of the second series in the group Y on transformations of the first series in the group X (and a constant).



The following transformations of the series are available for the bivariate fit:

None	y	x
Logarithmic	$\log(y)$	$\log(x)$
Inverse	$1/y$	$1/x$
Power	y^a	x^b
Box-Cox	$(y^a - 1)/a$	$(x^b - 1)/b$
Polynomial	—	$1, x, x^2, \ldots, x^b$

where you specify the parameters a and b in the edit field. Note that the Box-Cox transformation with parameter zero is the same as the log transformation.

- If any of the transformed values are not available, EViews returns an error message. For example, if you take logs of negative values, noninteger powers of nonpositive values, or inverses of zeros, EViews will stop processing and issue an error message.

- If you specify a high-order polynomial, EViews may be forced to drop some of the high order terms to avoid collinearity.

When you click **OK**, EViews displays a scatter diagram of the series together with a line connecting the fitted values from the regression. You may optionally save the fitted values as a series. Type a name for the fitted series in the **Fitted Y series** edit field.

Robustness Iterations

The least squares method is very sensitive to the presence of even a few outlying observations. The **Robustness Iterations** option carries out a form of weighted least squares where outlying observations are given relatively less weight in estimating the coefficients of the regression.

For any given transformation of the series, the **Robustness Iteration** option carries out robust fitting with bisquare weights. Robust fitting estimates the parameters a, b to minimize the weighted sum of squared residuals,

$$\sum_{i=1}^{N} r_i (y_i - a - x_i b)^2 \tag{13.7}$$

where y_i and x_i are the transformed series and the bisquare robustness weights r are given by,

$$r = \begin{cases} (1 - e_i^2/(36m^2))^2 & \text{for } |e_i/6m| < 1 \\ 0 & \text{otherwise} \end{cases} \tag{13.8}$$

where $e_i = y_i - a - x_i b$ is the residual from the previous iteration (the first iteration weights are determined by the OLS residuals), and m is the median of $|e_i|$. Observations with large residuals (outliers) are given small weights when forming the weighted sum of squared residuals.

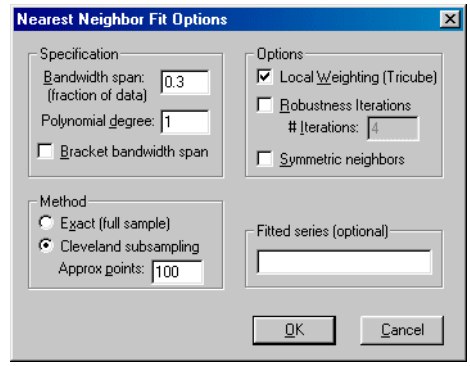
To choose robustness iterations, click on the check box for **Robustness Iterations** and specify an integer for the number of iterations.

See Cleveland (1993) for additional discussion.

Scatter with Nearest Neighbor Fit

This view displays local polynomial regressions with bandwidth based on nearest neighbors. Briefly, for each data point in a sample, we fit a locally weighted polynomial regression. It is a local regression since we use only the subset of observations which lie in a neighborhood of the point to fit the regression model; it may be weighted so that observations further from the given data point are given less weight.

This class of regressions includes the popular *Loess* (also known as *Lowess*) techniques described by Cleveland (1993, 1994). Additional discussion of these techniques may be found in Fan and Gijbels (1996), and in Chambers, Cleveland, Kleiner, Tukey (1983).



Method

You should choose between computing the local regression at each data point in the sample, or using a subsample of data points.

- **Exact (full sample)** fits a local regression at every data point in the sample.
- **Cleveland subsampling** performs the local regression at only a subset of points. You should provide the size of the subsample M in the edit box.

  The number of points at which the local regressions are computed is approximately equal to M. The actual number of points will depend on the distribution of the explanatory variable.

Since the exact method computes a regression at every data point in the sample, it may be quite time consuming when applied to large samples. For samples with over 100 observations, you may wish to consider subsampling.

The idea behind subsampling is that the local regression computed at two adjacent points should differ by only a small amount. Cleveland subsampling provides an adaptive algorithm for skipping nearby points in such a way that the subsample includes all of the representative values of the regressor.

It is worth emphasizing that at each point in the subsample, EViews uses the entire sample in determining the neighborhood of points. Thus, each regression in the Cleveland subsample corresponds to an equivalent regression in the exact computation. For large data sets, the computational savings are substantial, with very little loss of information.

Specification

For each point in the sample selected by the **Method** option, we compute the fitted value by running a local regression using data around that point. The **Specification** option determines the rules employed in identifying the observations to be included in each local regression, and the functional form used for the regression.

Bandwidth span determines which observations should be included in the local regressions. You should specify a number α between 0 and 1. The span controls the smoothness of the local fit; a larger fraction α gives a smoother fit. The fraction α instructs EViews to

include the $\lfloor \alpha N \rfloor$ observations nearest to the given point, where $\lfloor \alpha N \rfloor$ is 100α % of the total sample size, truncated to an integer.

Note that this standard definition of nearest neighbors implies that the number of points need not be symmetric around the point being evaluated. If desired, you can force symmetry by selecting the **Symmetric neighbors** option.

Polynomial degree specifies the degree of polynomial to fit in each local regression.

If you mark the **Bracket bandwidth span** option, EViews displays three nearest neighbor fits with spans of 0.5α, α, and 1.5α.

Other Options

Local Weighting (Tricube) weights the observations of each local regression. The weighted regression minimizes the weighted sum of squared residuals

$$\sum_{i=1}^{N} w_i (y_i - a - x_i b_1 - x_i^2 b_2 - \ldots - x_i^k b_k) . \tag{13.9}$$

The tricube weights w are given by

$$w_i = \begin{cases} \left(1 - \left|\frac{d_i}{d(\lfloor \alpha N \rfloor)}\right|^3\right)^3 & \text{for } \left|\frac{d_i}{d(\lfloor \alpha N \rfloor)}\right| < 1 \\ 0 & \text{otherwise} \end{cases} \tag{13.10}$$

where $d_i = |x_i - x|$ and $d(\lfloor \alpha N \rfloor)$ is the $\lfloor \alpha N \rfloor$-th smallest such distance. Observations that are relatively far from the point being evaluated get small weights in the sum of squared residuals. If you turn this option off, each local regression will be unweighted with $w_i = 1$ for all i.

Robustness Iterations iterates the local regressions by adjusting the weights to down-weight outlier observations. The initial fit is obtained using weights w_i, where w_i is tricube if you choose **Local Weighting** and 1 otherwise. The residuals e_i from the initial fit are used to compute the robustness bisquare weights r_i as given on (p. 402). In the second iteration, the local fit is obtained using weights $w_i r_i$. We repeat this process for the user specified number of iterations, where at each iteration the robustness weights r_i are recomputed using the residuals from the last iteration.

Symmetric Neighbors forces the local regression to include the same number of observations to the left and to the right of the point being evaluated. This approach violates the definition, though not the spirit, of nearest neighbor regression.

To save the fitted values as a series; type a name in the **Fitted series** field box. If you have specified subsampling, EViews will linearly interpolate to find the fitted value of y for the

actual value of x. If you have marked the **Bracket bandwidth span** option, EViews saves three series with _L, _M, _H appended to the name, each corresponding to bandwidths of 0.5α, α, and 1.5α, respectively.

Note that Loess is a special case of nearest neighbor fit, with a polynomial of degree 1, and local tricube weighting. The default EViews options are set to provide Loess estimation.

Scatter with Kernel Fit

This view displays fits of local polynomial kernel regressions of the second series in the group Y on the first series in the group X. Both the nearest neighbor fit, described above, and the kernel fit are nonparametric regressions that fit local polynomials. The two differ in how they define "local" in the choice of bandwidth. The effective bandwidth in nearest neighbor regression varies, adapting to the observed distribution of the regressor. For the kernel fit, the bandwidth is fixed but the local observations are weighted according to a kernel function.

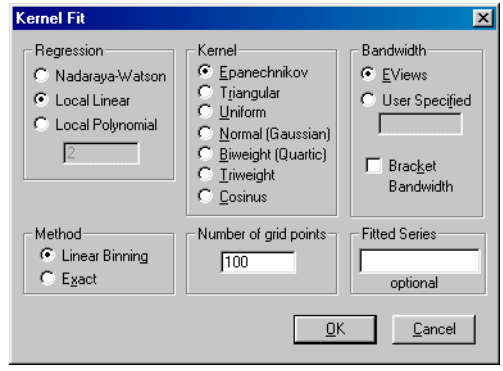
Extensive discussion may be found in Simonoff (1996), Hardle (1991), Fan and Gijbels (1996).

Local polynomial kernel regressions fit Y at each value x, by choosing the parameters β to minimize the weighted sum-of-squared residuals:

$$m(x) = \sum_{i=1}^{N} (Y_i - \beta_0 - \beta_1(x - X_i) + - \ldots - \beta_k(x - X_i)^k)^2 K\left(\frac{x - X_i}{h}\right) \quad (13.11)$$

where N is the number of observations, h is the bandwidth (or smoothing parameter), and K is a kernel function that integrates to one. Note that the minimizing estimates of β will differ for each x.

When you select the **Scatter with Kernel Fit** view, the **Kernel Fit** dialog appears.



You will need to specify the form of the local regression, the kernel, the bandwidth, and other options to control the fit procedure.

Regression

Specify the order of polynomial k to be fit at each data point. The **Nadaraya-Watson** option sets $k = 0$ and locally fits a constant at each x. **Local Linear** sets $k = 1$ at each x. For higher order polynomials, mark the **Local Polynomial** option and type in an integer in the field box to specify the order of the polynomial.

Kernel

The kernel is the function used to weight the observations in each local regression. EViews provides the option of selecting one of the following kernel functions:

Epanechnikov (default)	$\frac{3}{4}(1-u^2)I(\lvert u\rvert \le 1)$
Triangular	$(1-\lvert u\rvert)(I(\lvert u\rvert \le 1))$
Uniform (Rectangular)	$\frac{1}{2}(I(\lvert u\rvert \le 1))$
Normal (Gaussian)	$\frac{1}{\sqrt{2\pi}}\exp\left(-\frac{1}{2}u^2\right)$
Biweight (Quartic)	$\frac{15}{16}(1-u^2)^2 I(\lvert u\rvert \le 1)$
Triweight	$\frac{35}{32}(1-u^2)^3 I(\lvert u\rvert \le 1)$
Cosinus	$\frac{\pi}{4}\cos\left(\frac{\pi}{2}u\right)I(\lvert u\rvert \le 1)$

where u is the argument of the kernel function and I is the indicator function that takes a value of one, if its argument is true, and zero otherwise.

Bandwidth

The bandwidth h determines the weights to be applied to observations in each local regression. The larger the h, the smoother the fit. By default, **EViews** arbitrarily sets the bandwidth to:

$$h = 0.15(X_U - X_L) \tag{13.12}$$

where $(X_U - X_L)$ is the range of X.

For nearest neighbor bandwidths, see **Scatter with Nearest Neighbor Fit**.

To specify your own bandwidth, mark **User Specified** and enter a nonnegative number for the bandwidth in the edit box.

Bracket Bandwidth option fits three kernel regressions using bandwidths $0.5h$, h, and $1.5h$.

Number of grid points

You must specify the number of points M at which to evaluate the local polynomial regression. The default is $M = 100$ points; you can specify any integer in the field. Suppose the range of the series X is $[X_L, X_U]$. Then the polynomial is evaluated at M equispaced points:

$$x_i = X_L + i \cdot \left(\frac{X_U - X_L}{M} \right) \quad \text{for } i = 0, 1, \ldots M - 1 \tag{13.13}$$

Method

Given a number of evaluation points, EViews provides you with two additional computational options: exact computation and linear binning.

The **Exact** method performs a regression at each x_i, using all of the data points (X_j, Y_j), for $j = 1, 2, \ldots, N$. Since the exact method computes a regression at every grid point, it may be quite time consuming when applied to large samples. In these settings, you may wish to consider the linear binning method.

The **Linear Binning** method (Fan and Marron 1994) approximates the kernel regression by binning the raw data X_j fractionally to the two nearest evaluation points, prior to evaluating the kernel estimate. For large data sets, the computational savings may be substantial, with virtually no loss of precision.

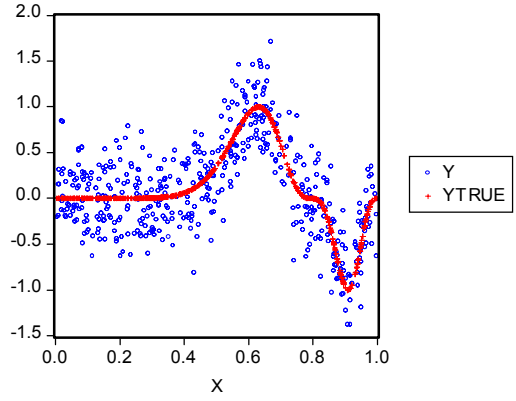
To save the fitted values as a series, type a name in the **Fitted Series** field box. EViews will save the fitted Y to the series, linearly interpolating points computed on the grid, to find the appropriate value. If you have marked the **Bracket Bandwidth** option, EViews saves three series with "_L", "_M", "_H" appended to the name, each corresponding to bandwidths 0.5α, α, and 1.5α, respectively.

Example

As an example, we estimate a bivariate relation for a simulated data set of the type used by Hardle (1991). The data were generated by:

```
scalar pi = @atan(1)*4
series x = rnd
series y = sin(2*pi*x^3)^3 + nrnd*(0.1^.5)
```

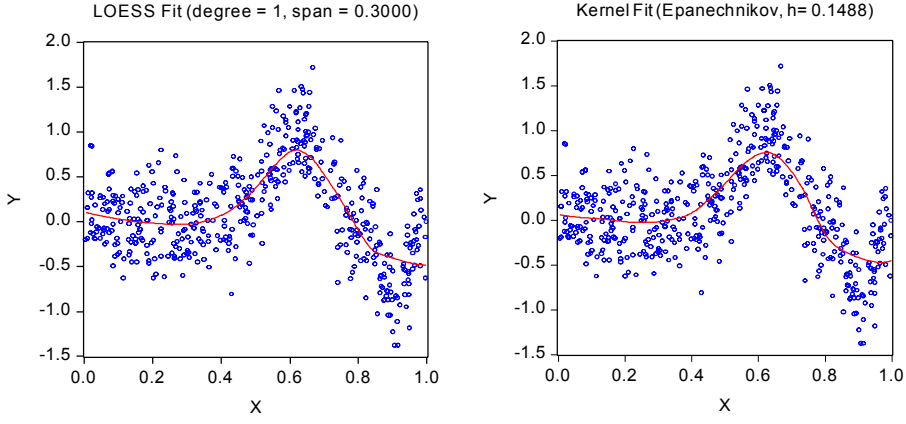
The simple scatter of Y and the "true" conditional mean of Y against X looks as follows:



The " + " shapes in the middle of the scatterplot trace out the "true" conditional mean of Y. Note that the true mean reaches a peak around $x = 0.6$, a valley around $x = 0.9$, and a saddle around $x = 0.8$.

To fit a nonparametric regression of Y on X, you first create a group containing the series Y and X. The order that you enter the series is important; the explanatory series variable must be the first series in the group. Highlight the series name X and then Y, double click in the highlighted area, select **Open Group**, and select **View/Graph/Scatter/Scatter with Nearest Neighbor Fit,** and repeat the procedure for **Scatter with Kernel Fit**.

The two fits, computed using the EViews default settings, are shown below:



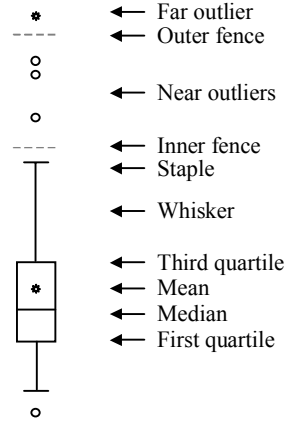
Both local regression lines seem to capture the peak, but the kernel fit is more sensitive to the upturn in the neighborhood of X = 1. Of course, the fitted lines change as we modify the options, particularly when we adjust the bandwidth h and window width α.

Boxplots

What is a boxplot?

A boxplot, also known as a box and whisker diagram, summarizes the distribution of a set of data by displaying the centering and spread of the data using a few primary elements.

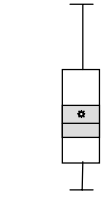
The box portion of a boxplot represents the first and third quartiles (middle 50 percent of the data). These two quartiles are collectively termed the *hinges*, and the difference between them represents the *interquartile range*, or IQR. The median is depicted using a line through the center of the box, while the mean is drawn using a symbol.



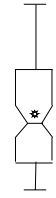
The *inner fences* are defined as the first quartile minus 1.5*IQR and the third quartile plus 1.5*IQR. The inner fences are not drawn, but graphic elements known as *whiskers* and *staples* show the values that are outside the first and third quartiles, but within the inner fences. The staple is a line drawn at the last data point within (or equal to) each of the inner fences. Whiskers are lines drawn from each hinge to the corresponding staple.

Data points outside the inner fence are known as *outliers*. To further characterize outliers, we define the *outer fences* as the first quartile minus 3.0*IQR and the third quartile plus 3.0*IQR. Data between the inner and outer fences are termed *near outliers*, and those outside the outer fence are referred to as *far outliers*. A data point lying on an outer fence is considered a near outlier.

A shaded region or notch may be added to the boxplot to display approximate confidence intervals for the median (under certain restrictive statistical assumptions). The bounds of the shaded or notched area are defined by the median +/- 1.57*IQR/$\sqrt{N}$,



Shaded boxplot



Notched boxplot

where N is the number of observations. Notching is useful in indicating whether two samples were drawn from populations with the same median; roughly speaking, if the notches of two boxes do not overlap, then the medians may be said to differ with 95% confidence. It is worth noting that in some cases, most likely involving small numbers of observations, the notches may be bigger than the boxes.

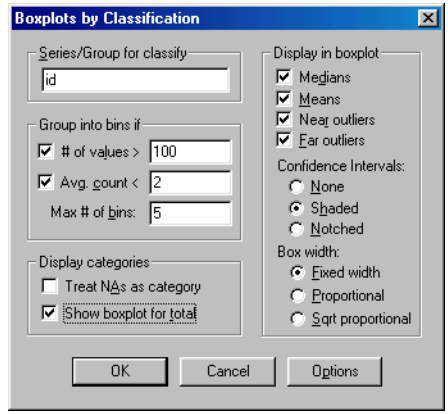
Boxplots are often drawn so that the widths of the boxes are uniform. Alternatively, the box widths can be varied as a measure of the sample size for each box, with widths drawn proportional to N, or proportional to the square root of N.

Creating a boxplot

The boxplot view can be created from a series for various subgroups of your sample, or from a group.

Boxplots by Classification

From a series, select **View/Descriptive Statistics/Boxplots by Classification…** to display the **Boxplots by Classification** dialog.



In the **Series/Group for classify** field, enter series or group names that define your subgroups. You may type more than one series or group name; separate each name by a space.

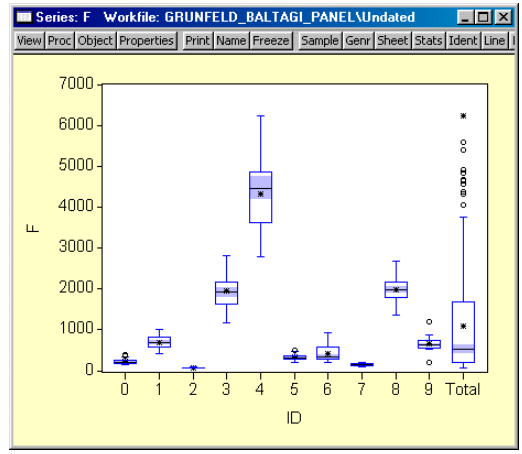
If the classification field is left blank, statistics will be calculated for the entire sample of observations, otherwise, descriptive statistics will be calculated for each unique value of the classification series (unless automatic binning is performed).

You may specify the NA handling, and the grouping options as described in "Stats by Classification" beginning on page 312. The **Show boxplot for total** option allows you to include a box of the summary statistics for (ungrouped) series the boxplot view.

The set of options provided in **Display in boxplot** allows you to customize the initialize appearance of the boxplot. By default, the **Fixed Width** boxplots will show **Medians**, **Means**, **Near outliers** and **Far outliers**, as well as **Shaded** confidence intervals for the medians. You need not make final decisions here, since all box display elements may subsequently be shown or hidden by modifying the resulting view.

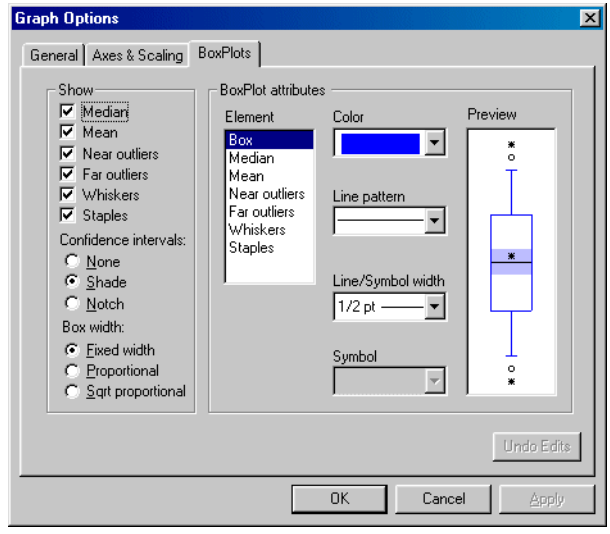
Lastly, the **Options** button may be used to open a dialog allowing you to customize the calculation of your quartiles.

Here we elect to display the boxplots for the series F categorized by the series ID. All of the settings are at the default values, except that we have selected the **Show boxplot for total** option.



Since there are 10 distinct values for ID, we display a separate boxplot for each value, showing the quartiles, means, outliers and median confidence intervals for F. Also displayed is a boxplot for labeled "Total", corresponding to a group made up of all of the observations for F in the current sample.

As noted above, you may elect to modify the characteristics of your boxplot display. Simply click anywhere in the main graph to bring up the **Graph Options** dialog.

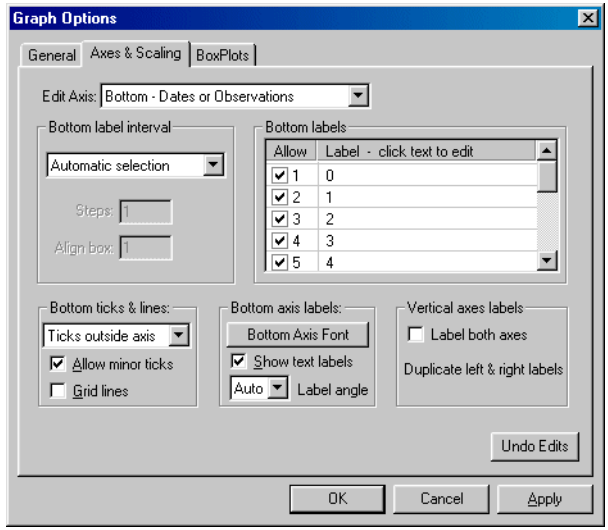


The left-hand side of the dialog repeats the display settings from the main boxplot create dialog. Here, you may choose to show or hide elements of the boxplot, and may modify the appearance of confidence intervals, or box widths.

In the right-hand portion of the dialog, you may customize individual elements of your graph. Simply select an element to customize by using the **Element** listbox, or by clicking on the depiction of a boxplot element in the **Preview** window, and then choose, as appropriate, the **Color**, **Line pattern**, **Line/Symbol width**, and **Symbol** type. Note that each boxplot element is represent by either a line or a symbol, and that the dialog will show the appropriate choice for the selected element.

The preview window will change to display the current settings for your graph. To keep the current settings, click on **Apply**. To revert to the original graph settings, click on **Undo Edits**.

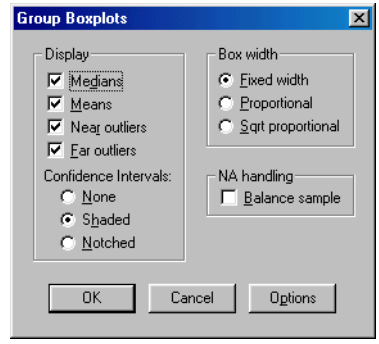
It is worth pointing out that the **Graph Options** dialog for boxplots does not include a **Type** or a **Legend** tab. Boxplots, like a number of other statistical graphs, do not allow you to change the graph type. In addition, boxplots use text objects in place of legends for labeling the graph.



To specify the axis labels for a boxplot, click on the **Axes & Scaling** tab, and select the **Bottom - Dates or Observations** axis in the **Edit Axis** combo box. You may use the listbox on the right side of the dialog to edit, display, or hide individual box labels. You may also change the box label font or display the labels at an angle.

Boxplots for a Group of Series

To display a boxplot for each series in a group, open a group window and select **View/ Descriptive Statistics/Boxplots...**. EViews will open the **Group Boxplots** dialog. You may use the dialog to customize the initialize appearance of the boxplot.



By default, the **Display** options are set to show all of the boxplot components: **Medians**, **Means**, **Near outliers** and **Far outliers**, and **Shaded** confidence intervals for the medians. You may elect to hide any of the basic elements by unchecking the corresponding checkbox, and you may use the radio buttons to change the display of the median confidence intervals: **None** or **Notched**.

The **Boxwidth** settings are, by default, set to show **Fixed Width** boxes, but you may elect to draw

boxes that have widths that are proportional to the number of observations in each series (**Proportional**), or proportional to the square root of the number of observations (**Sqrt proportional**).

You may elect to select the **Balance sample** checkbox, so that EViews will eliminate from the calculation sample those observations with a missing value for any of the series in the group. If this option is not selected, EViews will use the individual sample for each series.

Lastly, the **Options** button may be used to open a dialog allowing you to customize the calculation of your quartiles.

Chapter 14. Graphs, Tables, and Text Objects

EViews objects (series, groups, equations, and so on) display their views as graphs, tables, and text. You may, for example, display the descriptive statistics of a set of series or the regression output from an equation as a table, or the impulse responses from a VAR as a graph.

While object views may be customized in limited fashion, the customization is generally transitory. When you close the object and redisplay the view, most customized settings will be lost. Moreover, since views are often dynamic, when the underlying object changes, or the active sample changes, so do the object views.

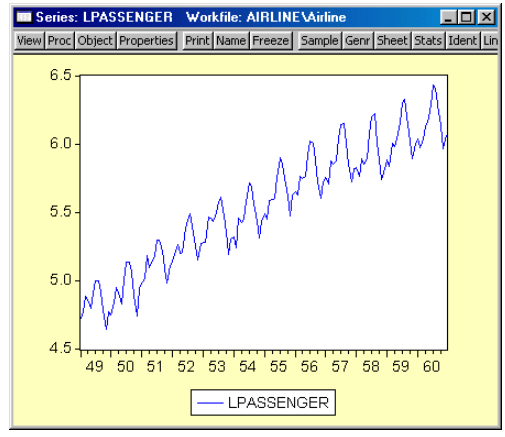
One may wish to preserve the current view, along with any customization, so that it does not change when the object changes. In EViews, this action is referred to as *freezing* the view. Freezing a view will create a new object containing a "snapshot" of the current contents of the view window. The type of object created varies with the original view: freezing a graphical view creates a graph object, freezing a tabular view creates a table object, and freezing a text view creates a text object.

Frozen views form the basis of most presentation output, and EViews provides tools for customizing the appearance of these objects. This chapter describes the options available for controlling the appearance of graph, table, and text objects.

Creating Graphs

Graph objects are usually created by freezing a view. Simply press the **Freeze** button in an object window which contains a graph view.

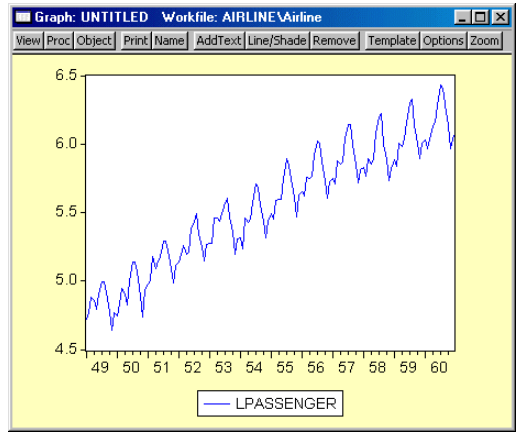
It is important to keep in mind the distinction between a graphical view of an object such as a series or a group, and a graph object created by freezing that view.



For example, suppose you wish to create a graph object containing a line graph of the series LPASSENGER. To display the line graph view of the series, select **View/Graph/Line** from the LPASSENGER series menu. Notice the "Series: LPASSENGER" designation in the window titlebar that shows this is a view of the series object.

You may customize this graph view in any of the ways described in "Customizing a Graph" on page 64 of the *Command and Programming Reference*, but these changes will be lost when the view is redrawn, *e.g.* when the object window is closed and reopened, when the workfile sample is modified, or when the data underlying the object are changed. If you would like to keep a customized graphical view, say for presentation purposes, you should create a graph object from the view.

To create a graph object from the view, click on the **Freeze** button. EViews will create an UNTITLED graph object containing a snapshot of the view.



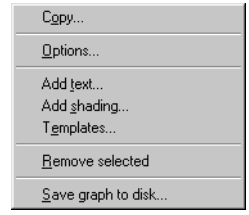
Here, the titlebar shows that we have an untitled graph object. The contents of the two windows are identical, since the graph object contains a copy of the contents of the original series view. Notice also that since we are working with a graph object, the menubar provides access to a new set of views and procedures which allow you to further modify the contents of the graph object.

As with other EViews objects, the UNTITLED graph will not be saved with the workfile. If you wish to store the frozen graph object in your workfile, you must name the graph object; press the **Name** button and provide a name.

You may also create a graph object by combining two or more existing named graph objects. Simply select all of the desired graphs and then double click on any one of the highlighted names. EViews will create a new, untitled graph, containing all of the selected graphs. An alternative method of combining graphs is to select **Quick/Show…** and enter the names of the graphs.

Modifying Graphs

A graph object is made up of a number of elements: the plot area, the axes, the graph legend, and one or more pieces of added text or shading. To select one of these elements for editing, simply click in the area associated with it. A blue box will appear around the selected element. Once you have made your selection, you can click and drag to move the element around the graph, or double click to bring up a dialog of options associated with the element.

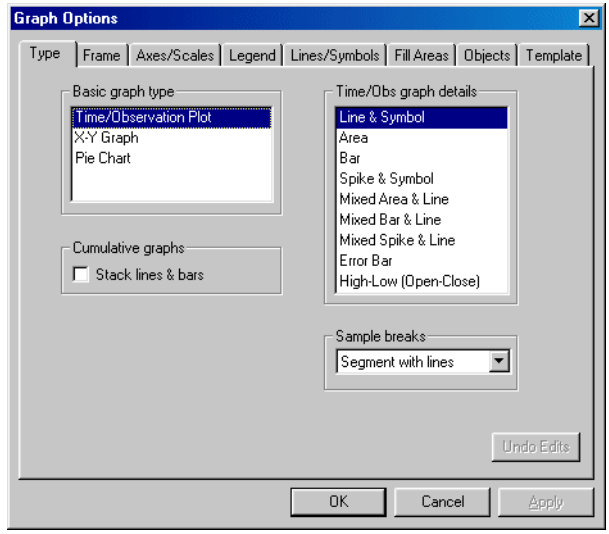


Alternatively, you may use the toolbar or the right mouse button menus to customize your graph. For example, clicking on the graph and then pressing the right mouse button brings up a menu containing tools for modifying and saving the graph.

The main options dialog may be opened by selecting **Options...** from the right mouse menu. You may also double click anywhere in the graph window to bring up the **Graph Options** tabbed dialog. If you double-click on an applicable graph element (the legend, axes, *etc.*), the dialog will open to the appropriate tab.

Graph Types

You may use the **Graph Options** dialog to customize your graph object. For most graphs, the **Type** tab allows you to change the graph type.



The listbox on the left-hand side of the **Type** tab provides access to the fundamental graph types. The listbox on the right will list the available graphs corresponding to the selected type. For example, if you select **Time/Observation Plot** on the left, the right-hand side listbox lists available time/observation graph types.

The fundamental graph types available depend on whether the graph involves a single series (or column of data) or more than one series (or column). For example, the **Mixed Area & Line**, **Mixed Bar & Line**, **Mixed Spike & Line**, **Error Bar**, and **High-Low (Open-Close)** types are only available for graphs containing multiple series or matrix columns.

Note also that a number of statistical graphs do not allow you to change the graph type. In this case, either the **Type** tab will be unavailable, or the **Basic graph type** will be set to **Special**.

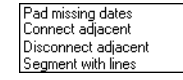
Most of the graph types are self-explanatory, but a few comments may prove useful.

If you select the **Line & Symbol** or **Spike & Symbol** type, you should use the **Lines & Symbols** tab to control the line pattern and/or symbol type. For area graphs, bar graphs, and pie charts, use the **Filled Areas** tab to control its appearance.

The **Error Bar** type is designed for displaying statistics with standard errors bands. This graph type shows a vertical error bar connecting the values for the first and second series. If the first series value is *below* the second series value, the bar will have outside half-lines. The (optional) third series is plotted as a symbol.

The **High-Low (Open-Close)** type displays up to four series. Data from the first two series (the high-low values) will be connected as a vertical line, the third series (the open value) is drawn as a left horizontal half-line, and the fourth series (the close value) is drawn as a right horizontal half-line. This graph type is commonly used to display daily stock price data.

The **Stack lines & bars** option plots the series so that each line represents the sum of all preceding series. In other words, each series value is the vertical distance between each successive line. Note that if some (but not all) of the series have missing values, the sum will be cumulated with missing values replaced by zeros.

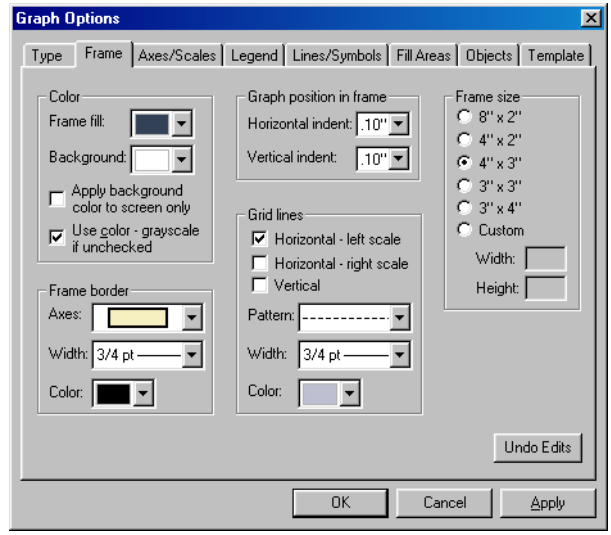


If your data includes sample breaks so that all of the observations in the graph are not consecutive, an additional pull-down menu will appear in the bottom right-hand portion of the tab, allowing you to select how breaks are handled in the graph. You may, for example, choose to **Pad the graph** so that missing observations appear as blanks, or you may elect to connect the non-consecutive graphs, with or without line segments showing the breaks (**Segment with lines**).

Once you have selected the graph type settings, click on **Apply** or **OK** to change the graph type. Selecting **OK** also closes the dialog.

Basic Graph Characteristics

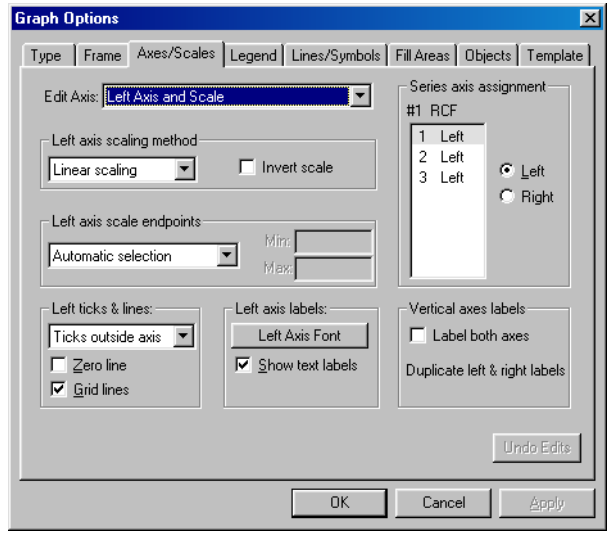
Underlying Graph Attributes



The **Frame** tab controls basic display characteristics of the graph, including color usage, framing style, indent position, grid lines.

You can also use this tab to control the aspect ratio of your graph using the predefined ratios, or you can input a custom set of dimensions. Note that the values are displayed in "virtual inches".

Bear in mind that if have previously added text in the graph with user specified (absolute) position, changing the graph frame size may change the *relative* position of the text in the graph.



To change or edit axes, select the **Axes & Scaling** tab. Depending on its type, a graph can have up to four axes: left, bottom, right, and top. Each series is assigned an axis as displayed in the upper right listbox.

You may change the assigned axis by first highlighting the series and then clicking on one of the available axis buttons. For example, to plot several series with a common scale, you should assign all series to the same axis. To plot two series with a dual left-right scale, assign different axes to the two series.

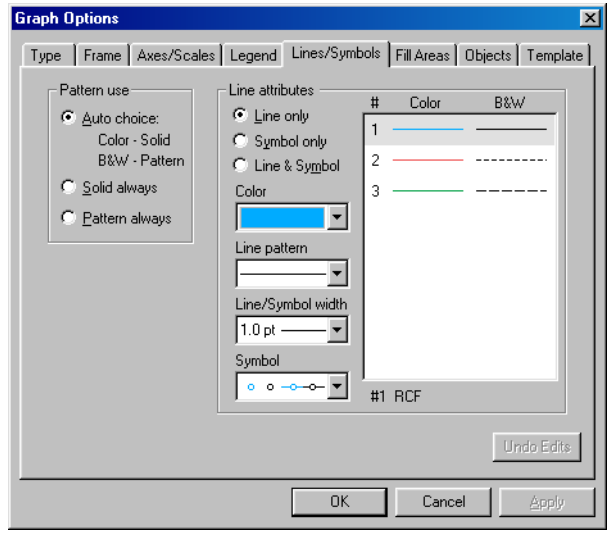
To edit characteristics of an axis, select the desired axis from the drop down menu at the top of the dialog; the left/right axes may be customized for all graphs. For the **Time/ Observation Plot** type, you may edit the bottom axis to control how the dates/observations are labeled. Alternately, for XY graphs, the bottom/top axes may be edited to control the appearance of the data scale.

To edit the graph legend characteristics, select the **Legend** tab. Note that if you place the legend using user specified (absolute) positions, the relative position of the legend may change when you change the graph frame size.

Data Display Attributes (Lines, Symbols, Filled Areas)

The **Lines/Symbols** tab provides you with control over the drawing of all lines and symbols corresponding to the data in your graph.

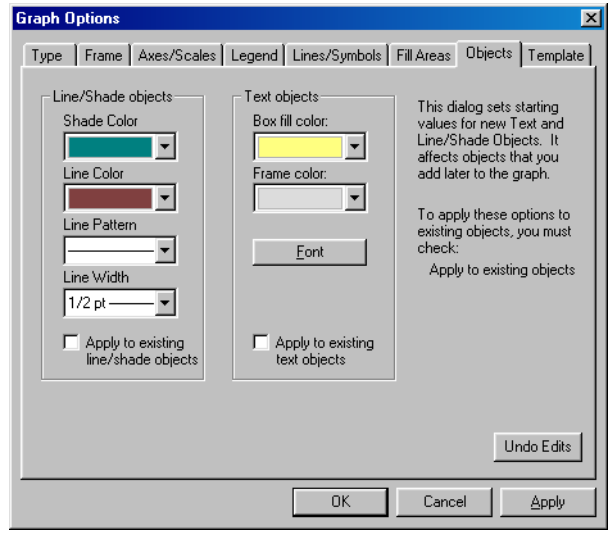
You may choose to display lines, symbols, or both, and you can customize the color, width, pattern, and symbol usage. See "Use of color with lines and filled areas" on page 65 of the *Command and Programming Reference*.



The current line and symbol settings will be displayed in the listbox on the right hand side of the dialog. Once you make your choices, click on **Apply** to see the effect of the new settings.

The **Filled Areas** tab allows you to control the display characteristics of your area, bar, or pie graph. Here, you may customize the color, shading, and labeling of the graph elements.

Added Text, Line, and Shade Attribute Defaults



The **Objects** tab allows you to control the default characteristics of new text, shade, or line drawing objects later added to the graph (see "Adding and Editing Text" on page 421 and "Adding Lines and Shades" on page 423).

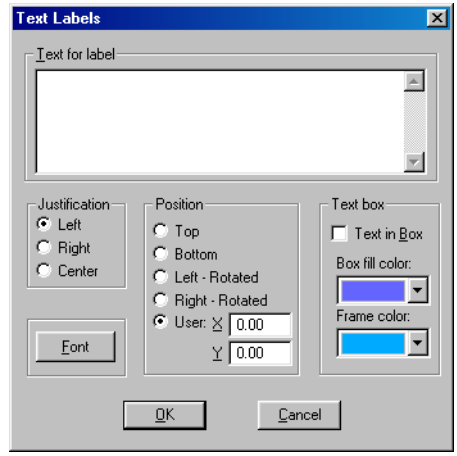
You may select colors for the shade, line, box, or text box frame, as well as line patterns and widths, and text fonts and font characteristics.

By default, when you apply these changes to the graph object options, EViews will update the default settings in the graph, and will use these settings when creating new line, shade, or text objects. Any existing lines, shades or text in the graph will not be updated. If you wish to modify the existing to use the new settings, you must check the **Apply to existing line/shade objects** and **Apply to existing text objects** boxes prior to clicking on the **Apply** button.

Adding and Editing Text

You can customize a graph by adding one or more lines of text anywhere in the graph. This can be useful for labeling a particular observation or period, or for adding titles or remarks to the graph. To add new text, simply click on the **AddText** button in the toolbar or select **Proc/Add text…**

To modify an existing text object, simply double click on the object. The **Text Labels** dialog will be displayed.

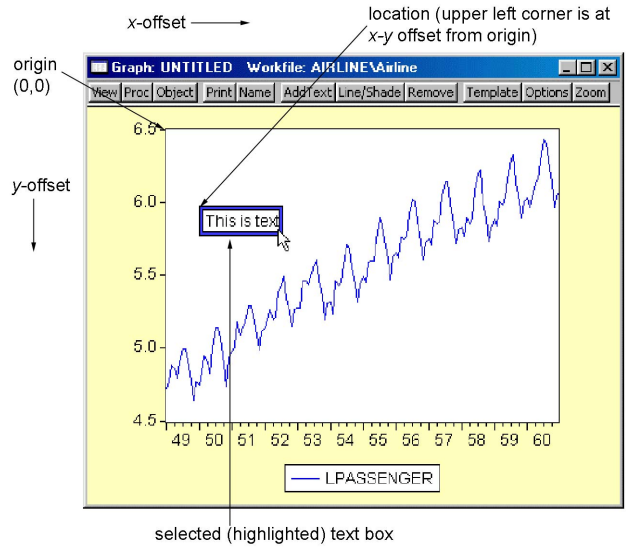


Enter the text you wish to display in the large edit field. Spacing and capitalization (upper and lower case letters) will be preserved. If you want to enter more than one line, press the **Enter** key after each line.

- The **Justification** options determine how multiple lines will be aligned relative to each other.
- **Font** allows you to select a font and font characteristics for the text.
- **Text in Box** encloses the text in a box.
- **Box fill color** controls the color of the area inside the text box.
- **Frame color** controls the color of the frame of the text box.

The first four options in **Position** place the text at the indicated (relative) position outside the graph. You can also place the text by specifying its coordinates. Coordinates are set in virtual inches, with the origin at the upper left-hand corner of the graph.

The X-axis position increases as you move to the right of the origin, while the Y-axis increases as you move down from the origin. The default sizes, which are expressed in virtual inches, are taken from the global options, with the exception of scatter diagrams always default to 3×3 virtual inches.



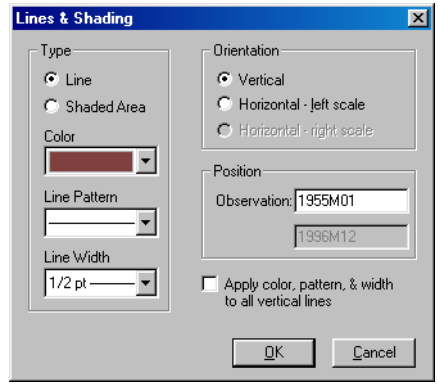
Consider, for example, a graph with a size of 4×3 virtual inches. For this graph, the X = 4, Y = 3 position refers to the lower right

hand corner of the graph. Labels will be placed with the upper left-hand corner of the enclosing box at the specified coordinate.

You can change the position of text added to the graph by selecting the text box and dragging it to the position you choose. After dragging to the desired position, you may double click on the text to bring up the **Text Labels** dialog to check the coordinates of that position or to make changes to the text. Note that if you specify the text position using coordinates, the *relative* position of the text may change when you change the graph frame size.

Adding Lines and Shades

You may draw lines or add a shaded area to the graph. From a graph object, click on the **Lines/Shade** button in the toolbar or select **Proc/Add shading…**. The **Lines & Shading** dialog will appear:



Select whether you want to draw a line or add a shaded area, and enter the appropriate information to position the line or shaded area horizontally or vertically. If you select **Vertical**, EViews will prompt you to position the line or shaded area at a given observation. If you select **Horizontal**, you must provide a data value at which to draw the line or shaded area.

You should also use this dialog to choose a line pattern, width, and color for the line or shaded area, using the drop down menus.

If you check the **Apply color...** checkbox, EViews will update all of the existing lines or shades of the specified type in the graph.

To modify a single existing line or shaded area, simply double click on it to bring up the dialog.

Removing Graph Elements

To remove a graph element, simply select the element and press the **Delete** key. Alternately, you may select the element and then press the **Remove** button on the graph toolbar. For example, to remove text that you have placed on the graph, click on the text. A border will appear around the text. Press **Delete** or click on the **Remove** button to delete the text.

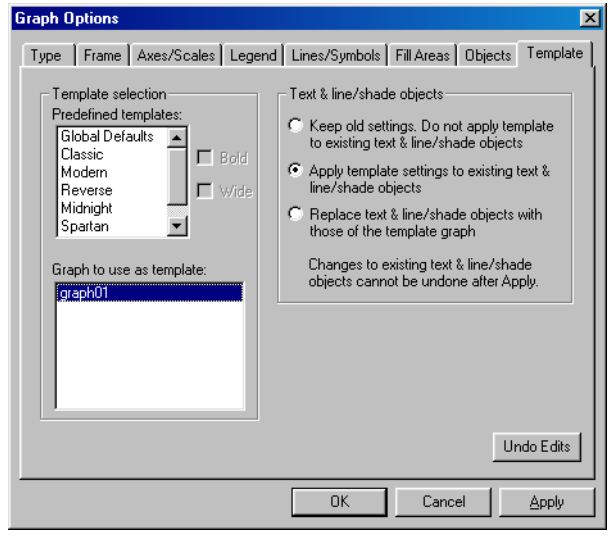
The same method may be applied to legends, scales, lines, or shading which have been added to the graph.

Graph Templates

Having put a lot of effort into getting a graph to look just the way you want it, you may want to use the same options in another graph. EViews allows you to use any named graph as a *template* for a new or existing graph. You may think of a template as a graph style that can be applied to other graphs.

In addition, EViews provides a set of predefined templates that you may use to customize the graph. These predefined templates are not associated with objects in the workfile, and you are always available. The EViews templates provide easy-to-use examples of graph customization that may be applied to any graph. You may also find it useful to use the predefined templates as a foundation for your own graph template creation.

To update a graph using a template, double click on the graph area to display the **Graph Options** dialog, and click on the **Template** tab. Alternatively, you may right mouse click, and select **Template...** to open the desired tab of the dialog.



On the left-hand side of the dialog you will first select your template. The upper list box contains a list of the EViews predefined templates. The lower list box contains a list of all of the named graphs in the current workfile page. Here, we have selected the graph object GRAPH01 for use as our graph template.

If instead, you select one of the predefined templates, you will be given the choice of applying the **Bold** or **Wide** modifiers to the base template. As the name suggests, the **Bold** modifier changes the settings in the template so that lines and symbols are bolder (thicker, and larger) and adjusts other characteristics of the graph, such as the frame, to match. The **Wide** modifier changes the aspect ratio of the graph so that the horizontal to vertical ratio is increased.

You may reset the dialog by clicking on the **Undo Edits** button prior to clicking on **Apply**. When you click on the **Apply** button, EViews will immediately update all of the basic graph settings described in "Basic Graph Characteristics" on page 419, including graph size and aspect ratio, frame color and width, graph background color, grid line options, and

line, symbol, and filled area settings. Once applied, these changes cannot be undone automatically.

In contrast to the basic graph settings which are always updated when you click on **Apply**, the impact on the characteristics of existing text, line, and shade objects in the graph ("Added Text, Line, and Shade Attribute Defaults" on page 421) is controlled by the choices on the right-hand side of the dialog. There are three possibilities:

- **Keep old settings –** instructs EViews to use the text, line, and shade attributes in the template or template graph only for the purpose of updating the default settings in the graph. If you select this option and select **Apply**, *subsequently* added text, line, and shades will use the updated settings, but existing objects will retain their existing characterisitcs.
- **Apply template settings to existing text & line/shade objects** – will update both the settings for existing text, line, and shade objects, and the defaults used for newly added objects.
- **Replace text & line/shade objects with those of the template graph –** will first remove any added text label, line, or shading objects in the existing graph, and then copy to the graph any such objects in the template.

Setting Global Defaults

You may change the default settings for any of these options by selecting **Options/Graphics Defaults...** from the main EViews menu. Any new graph views or objects will use the graphic options as the default settings.

Multiple Graphs

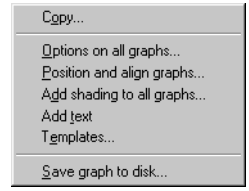
Some views are made up of multiple graphs. Like single graph views, these may be turned into graph objects by freezing. For example, the impulse response view of a VAR can display multiple graphs in a single view.

You may also create a graph object containing multiple graphs by combining existing named graph objects. Simply select the desired graphs and then double click on any one of the highlighted names. An alternative method of combining graphs is to select **Quick/Show…** and enter the names of the graphs.

There are two ways to work with a multiple graph. You may change the settings for the multiple graph as a whole, or you may work with an individual graph component of the multiple graph.

Working With Multiple Graphs

EViews makes it easy to work with all of the graphs in a multiple graph. Simply select **Proc** from the graph menu and EViews will display a menu prompting you for additional choices.



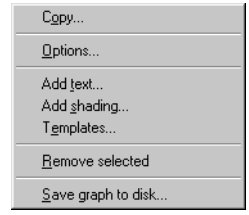
These menu items set options that apply to all graphs in the graph object.

- To set a common graph attribute to all graphs, select **Options on all graphs…**. Alternately, you may click anywhere in the background area of the multiple graph. After selecting the desired options, check the **Apply page to all graphs** checkbox at the bottom of the tab.
- While each single graph in a multiple graph can be freely positioned by dragging the graph, you may wish to globally align graphs in columns and control the overall spacing between graphs. To globally position your graphs, select **Position and align graphs...**
- If all of your graphs share a common axis, you can draw lines or add shading to each graph in the object, by selecting **Add shading to all graphs…**.
- Selecting **Add text…** allows you to annotate your multiple graph. Note that adding an item to the multiple graph differs from adding it to an individual graph since it will not move with the individual graph.
- Selecting **Template...** allows you to apply a template graph to each individual graph in your multiple graph or to reset the graph to use the global defaults.
- **Save graph to disk...** brings up the **File Save** dialog, as described in "Saving Graphs to a File" on page 429.

There is a shortcut method if you merely wish to set the options for all of your graphs. Simply double click anywhere in the background area of your graph, and EViews will open the multiple **Graph Options** dialog.

Working with Individual Graphs

You may change the options for an individual graph in the usual fashion by double clicking on the graph to display the options dialog.

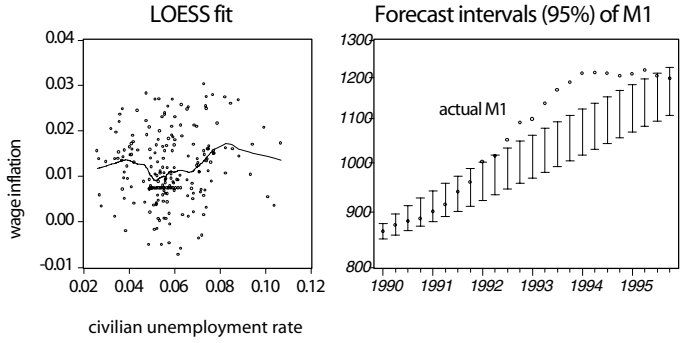


You can also perform various operations on individual graphs. Click on the graph and EViews will confirm the selection by surrounding the individual graph with a blue border. Select **Proc** or right mouse click to display a menu that allows you to set options, add text or shading, apply a template or reset options to the global defaults, to remove the selected graph, or to save the graph to disk. Deleting the selected graph can also be performed by pressing the **Remove** button on the graph toolbar or the **Delete** key.

In addition, you can place each individual graph at an arbitrary position by simply dragging the individual graph to the desired location.

An Example

Here is an example of an annotated and customized combined graph created in EViews and copied-and-pasted into a word processor:

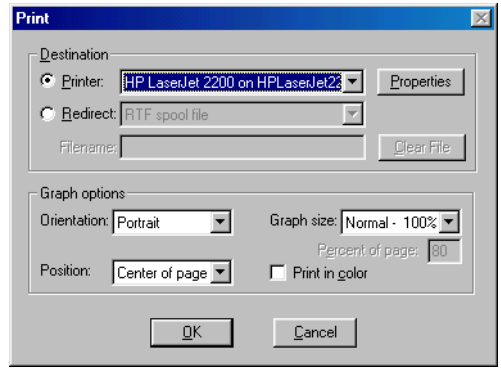


See "Copying Tables to the Clipboard" on page 437 for additional discussion of exporting graphs via copy-and-paste.

Printing Graphs

Clicking on the **Print** button on the graph view or graph object window toolbar will open the **Print** dialog, allowing you to override the various global settings for graph printing.

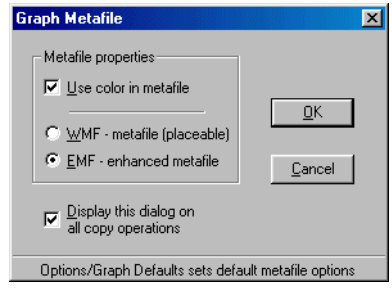
Most of the options are self-explanatory. If you wish to print your graph in color using your color printer, make certain that the **Print in color** box is checked. Conversely, if you are printing to a black and white printer, you should make certain that this box is not checked so that EViews will substitute line patterns for colors. See "Print Setup" on page 943 for additional details.



Copying Graphs to the Clipboard

You can incorporate an EViews graph view or object directly into a document in your Windows word processor. First, you should activate the object window containing the graph you wish to move by clicking anywhere in the window (the titlebar of the object window should change to a bright color). Then click on **Edit/Copy** on the EViews main menu; the **Graph Metafile** dialog box appears. The default settings in this dialog are taken from the global defaults.

You can copy the graph to the Windows clipboard in Windows metafile (WMF) or enhanced metafile (EMF) formats. You can request that the graph be in color and that its lines be in bold. We recommend that you copy graphs in black-and-white unless you will be printing to a color printer.



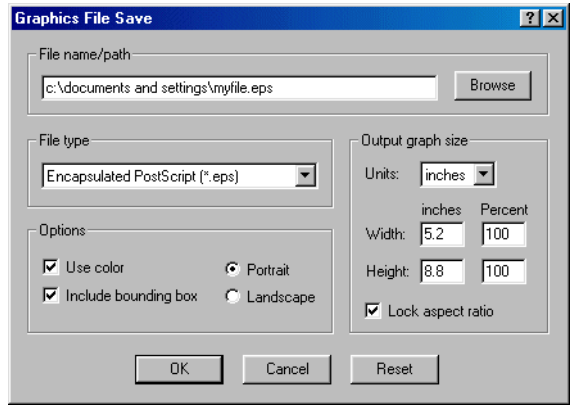
Once you copy a graph to the clipboard, you may then switch to your word processor and paste the graph into your document. Standard programs such as Microsoft Word will give you a graph which can be sized, positioned, and modified within the program. You can also paste graphs into drawing programs, and make further modifications before pasting into your word processor or other software.

You may choose to hide this copy dialog for subsequent operations by unchecking the **Display this dialog...** box. Copying will then always use the default settings, without prompting. If you wish to change the default settings, or to turn on or off the display of the copy dialog, you may go to the **Exporting** tab of the global Graph options (**Options/Graphics Defaults...**).

Saving Graphs to a File

When working with graph objects, EViews also allows you to create a metafile or PostScript file.

You may, at any time, right mouse click or click on **Proc**, then select **Save graph to disk...** to bring up the dialog.



In the top portion of the dialog, you should provide the name of the file you wish to create. EViews will automatically append an extension of the proper type to the name (here, ".EPS" since we are saving an Encapsulated PostScript file).

Next, select the **File type**, and any options associated with the output type. You may select **Metafile - placeable**, **Enhanced Metafile**, or **Encapsulated PostScript**. You may elect to save the graph in color or not, and, for PostScript files, include a bounding box or choose the graph orientation.

Lastly, you should select the **Output graph size**. The size may be specified in inches, centimeters, printer points, or picas. If the **Lock aspect ratio** checkbox is selected, changes to the **Width** or the **Height** will generate corresponding changes in the other dimension. If you wish to scale your graph in a non-proportionate fashion, you should uncheck this box.

The default graph file saving options may be set in the global options dialog by selecting **Options/Graphics Defaults...**. (see "Graphics Defaults" on page 941).

Graph Commands

For those of you who wish to automate these procedures, for example to produce a regular report, EViews allows you to perform extensive graph customization from the command line or using programs. See Chapter 5, "Working with Graphs", on page 59 and "Graph" (p. 161) in the *Command and Programming Reference* for additional details.

Creating Tables

In EViews, a table may be an object view or a table object. Table views are object views that contain formatted text or numbers that are aligned in columns and rows. Examples of

table views are the spreadsheet views of a series and the estimation output views of an equation. There are a limited set of customizations that are available for table views.

A table object is an independent object that contains formatted text or numbers. Table objects may be created directly, by issuing a table declaration, or indirectly, by freezing a table view. As with graph objects, table objects are "not live" in the sense that they do not reflect the current contents of the underlying object, but are based instead upon the contents of the object at the time the object was frozen. Table objects also allow for a full set of customizations.

While many of the features described here apply to both table views and table objects, the remaining discussion focuses on customization of table objects. Working with table views is described elsewhere (see, for example "Changing the Spreadsheet Display" on page 88).

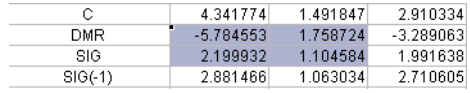
Table Basics

The most basic operations in a table involve selecting cells and editing cell values.

Selecting Cells

Selecting one or more cells is one of the most common tasks in working with table views and table objects. For the most part, you will find that cell selection works as it does everywhere else in Windows, but a brief review may prove useful.

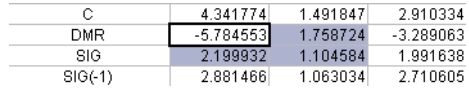
The simplest selection is for a single cell. Simply click on the cell you wish to select. If the table is not in edit mode, the cell will be shaded. If the table is in edit mode, the cell will be surrounded by a black border, and the contents of the cell will be displayed in the table edit window.



C	4.341774	1.491847	2.910334
DMR	-5.784553	1.758724	-3.289063
SIG	2.199932	1.104584	1.991638
SIG(-1)	2.881466	1.063034	2.710605

For the selection of multiple cells, EViews uses the concept of an *anchor cell* to determine a selection region. The anchor cell is used to mark the start of the selection region and is used to indicate how a selection will change as you move the mouse or use keystrokes.

When edit mode is off, the anchor cell is marked as the cell with the black square in one of the four corners of the cell. When edit mode is on, the anchor cell is marked with a black border around the cell. You may toggle between edit mode on and edit mode off by clicking on the **Edit +/-** button on the object toolbar, or alternately, by right mouse clicking and selecting **Edit +/-**.



C	4.341774	1.491847	2.910334
DMR	-5.784553	1.758724	-3.289063
SIG	2.199932	1.104584	1.991638
SIG(-1)	2.881466	1.063034	2.710605

The easiest way to highlight a region is to (left) click in a cell to set an anchor point, then, while holding down the mouse but-

ton, move the mouse to select additional cells. In addition, cell selection shortcuts allow you to select rows and columns by clicking on row and column headers, and to select rectangular regions by clicking on a cell to set an anchor cell, then SHIFT-click to select the rectangular region defined by the anchor and ending cells. You may enter CTRL-A to select all of the cells in a table.

Some of the more frequently used selection tools include:

To select	Action
Text in a cell	If edit mode is turned on, select the cell, double-click in it, and then select the text in the cell. Or select the cell and then select the text in the edit field.
A single cell	Click the cell, or use the arrow keys to move the anchor cell.
A range of cells	Click the first cell of the range, and then drag to the last cell. Or click in a cell to set the anchor, then SHIFT-click in the last cell you wish to include. Or set the anchor, and then SHIFT and press the arrow keys until desired cells are selected.
All cells in a table	Click the corner cell shared by the column and row header (the corner cell is not visible in some output views). Or press CTRL + A.
An entire row	Click the row heading.
An entire column	Click the column heading.
Adjacent rows or columns	Click and drag across the row or column headers. Or select the first row or column; then hold down SHIFT and click the last row or column heading.
More or fewer cells than the current selection	Hold down SHIFT and click the last cell you want to include in the new selection. The rectangular range between the active cell and the cell you click becomes the new selection. Or hold down SHIFT and press the arrow keys until selection is correct.

Note that row and column header selection is not always available in table views since the headers are not always displayed. For example, the estimation output view of an equation is a table that does not contain header lines. Freezing the view creates a table object that allows for cell selection using the visible headers.

Editing Cell Values

To enter or change the data in a table, you must first display the table edit window by enabling edit mode, and selecting a cell to be modified. Here, we see a table that is in edit mode, with the contents of the A1 cell displayed in the edit window just below the toolbar.

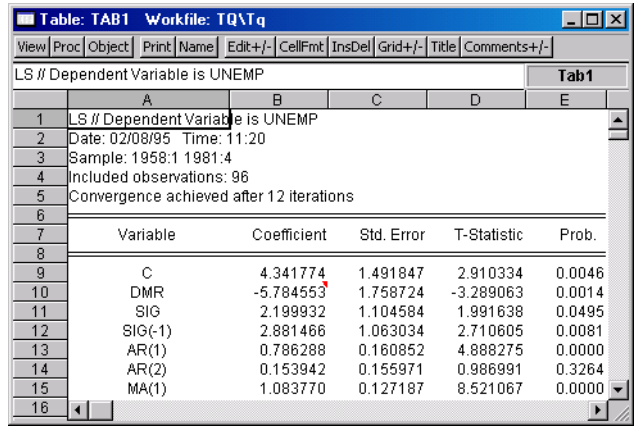


Table: TAB1 Workfile: TQ\Tq
View Proc Object Print Name Edit+/- CellFmt InsDel Grid+/- Title Comments+/-
LS // Dependent Variable is UNEMP
Tab1
A B C D E
1 LS // Dependent Variable is UNEMP
2 Date: 02/08/95 Time: 11:20
3 Sample: 1958:1 1981:4
4 Included observations: 96
5 Convergence achieved after 12 iterations
6
7 Variable Coefficient Std. Error T-Statistic Prob.
8
9 C 4.341774 1.491847 2.910334 0.0046
10 DMR -5.784553 1.758724 -3.289063 0.0014
11 SIG 2.199932 1.104584 1.991638 0.0495
12 SIG(-1) 2.881466 1.063034 2.710605 0.0081
13 AR(1) 0.786288 0.160852 4.888275 0.0000
14 AR(2) 0.153942 0.155971 0.986991 0.3264
15 MA(1) 1.083770 0.127187 8.521067 0.0000
16

To modify the contents of the cell, simply type in the edit window. Alternately, you may double click in the cell to edit the contents. EViews will then allow you to edit the cell in place.

You may provide either alphanumeric or numeric input. If your text may be interpreted as a number, EViews will interpret the input and store the value as a number. Since the table value is stored as a number it may later be formatted using the numeric formatting tools. You may, for example, change the display of the number to scientific notation, or you may display numbers with 4 digits of precision (see "Content Formatting" on page 434).

Note that you may enter numeric expressions and have EViews evaluate them prior to placing them in the table. To evaluate a numeric expression into a cell, type " = " before the expression. For example, entering the text " = 4*5" will result in a cell value of "20". Entering an invalid numeric expression will set the cell to a numeric NA.

This latter example raises a minor issue associated with entering missing values into a table. If the text "NA" is entered into a table cell, the cell value will be set to the string "NA", not to the missing value NA. To enter a numeric missing value, you should enter the string " = NA" into the cell. We point out that the choice between entering the "NA" string or the NA value into a cell has consequences for auto-justification, or when saving values to a file.

Basic Table Customization

You may perform basic customization of a table object by attaching a title, by adding or hiding the grid lines, or by resizing the rows or columns.

Table Title

To add a header title to the top of a table object, you should select **Proc/Title...** from the table menu, or you may click on the **Title** button on the toolbar. EViews will display a dialog prompting you to enter your title. When you enter text in this dialog, EViews displays a header title at the top center of the table. Note that the table title is different from the table name, which provides the object name for the table in the workfile.

To remove the table title, display the title dialog, then delete the existing title.

Gridlines

To toggle on or off the grid marking the cells in the table object, click on the **Grid + /–** button on the table toolbar, or select **Proc/Grid + /-** from the main table menu.

Resizing Columns and Rows

Column widths may easily be resized in both table views and in table objects. Simply place your cursor over the separator lines in the column header. When the cursor changes to the two-sided arrow, click and drag the column separator until the column is the desired size. If you wish to resize more than one column to the same size, first select the columns you wish to resize, then drag a single column separator to the desired size. When you release the mouse button, all of the columns will be resized to the specified size.

Row heights may only be resized in table objects. Place your cursor over the separator lines in the row header and drag the separator until the row is the desired height. If you wish to resize more than one row, first select the rows you wish to resize, then drag a separator to the desired size. All of the rows will be resized to the specified size.

Double clicking a column/row edge in the header will resize the row or column to the minimum height or width required so that all of the data in that row or column is visible.

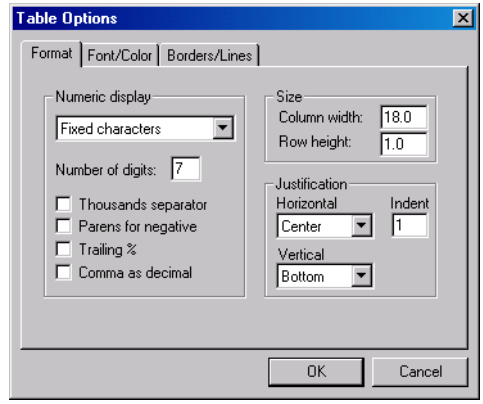
Customizing Table Cells

EViews provides considerable control over the appearance of table cells, allowing you to specify content formatting, justification, font face, size, and color, cell background color and borders. Cell merging and annotation are also supported.

Cell Formatting

You may select individual cells, ranges of cells, or the entire table, and apply various formatting tools to your selection. To format the contents of a set of cells, first make certain that the table is in edit mode. Next, select a cell region, then click on **CellFmt** in the toolbar, or right mouse click within the selected cell region and select **Cell Format...**. EViews will open the **Table Options** dialog containing three tabs: **Format**, **Font/Color**, and **Borders/Lines**.

Content Formatting



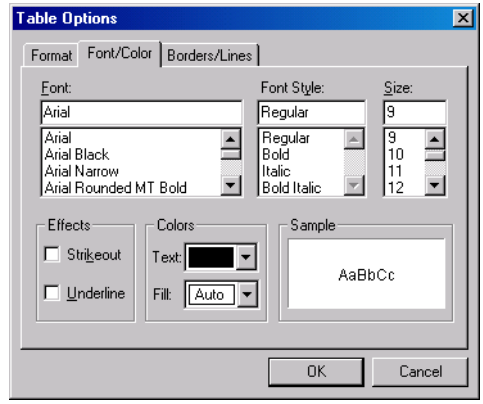
The **Format** tab allows you to apply display formats to the contents of cells in table objects. Formatting of table objects may be cell specific, so that each cell may contain its own format.

You may also modify the display of numeric values, set column widths and row heights, and specify the justification and indentation.

Bear in mind that changing the height of a cell changes the height of the entire row and changing the width of a cell changes the width of the column. Column widths are expressed in unit widths of a numeric character, where the character is based on the default font of the table at the time of creation. Row height is measured in unit heights of a numeric character in the default font.

For additional discussion of content and cell formatting, see the related discussion in "Changing the Spreadsheet Display" on page 88.

Fonts and Fill Color



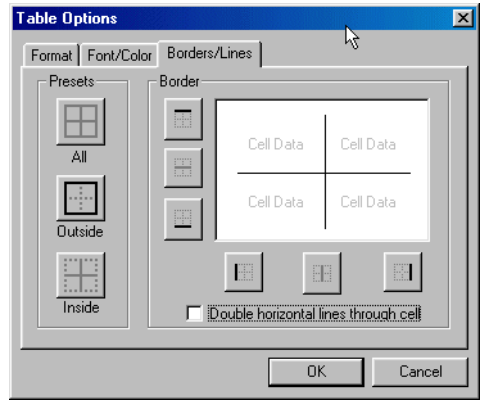
The **Font/Color** tab allows you to specify the font face, style, size and color for text in the specified cells. You may also add strikeout and underline effects to the font. This dialog may also be used to specify the background fill color for the selected cells.

Where possible, the **Sample** window displays a preview of the current settings for the selected cells. In cases where it is impossible to display a preview (the selected cells do not have the same fonts, text colors, or fill colors) the sample text will be displayed as gray text on a white background.

Note also that EViews uses the special keyword **Auto** to identify cases where the selection region contains more than one text or fill color. To apply new colors to all of the selected cells, simply select a **Text** or **Fill** color and click on **OK**.

Borders and Lines

The last tab, labeled **Borders/Lines** is used to specify borders and lines for the selected table cells.



Simply click on any of the **Presets** or **Border** buttons to turn on or off the drawing of borders for the selected cells, as depicted on the button. As you turn on and off border lines, both the buttons and the display on the right will change to reflect the current state of your selections. Note also that there is a checkbox allowing you to draw double horizontal lines through the selected cells.

It is worth noting that the appearance of the **Borders/Lines** page will differ slightly depending on whether your current selection contains a single cell or more than one row or column of cells. In this example, we see the dialog for a selection consisting of multiple rows and columns. There are three sets of buttons in the **Border** section for toggling both the row and column borders. The first and last buttons correspond to the outer borders, and the second button is used to set the between cell inner border.

If there were a single column in the selection region, the **Border** display would only show a single column of “Cell Data”, and would have only two buttons for modifying the outer vertical cell borders. Similarly, if there were a single row of cells, there would be a single row of “Cell Data”, and two buttons for modifying the outer horizontal cell borders.

Cell Annotation

Each cell of a table object is capable of containing a comment. Comments may be used to make notes on the contents of a cell without changing the appearance of the table, since they are hidden until the mouse cursor is placed over a cell containing a comment.

To add a comment, select the cell that is to contain the comment, then right mouse click and select **Insert Comment...** to open the **Insert Cell Comment** dialog. Enter the text for your comment, then click **OK**. To delete an existing comment, just remove the comment string from the dialog.

If comment mode is on, a cell containing a comment will be displayed with a small red triangle in its upper right-hand corner. When the cursor is placed over the cell, the comment will be displayed.

If comment mode is off, the red indicator will not be displayed, but the comment will still appear when the cursor is placed over the cell.

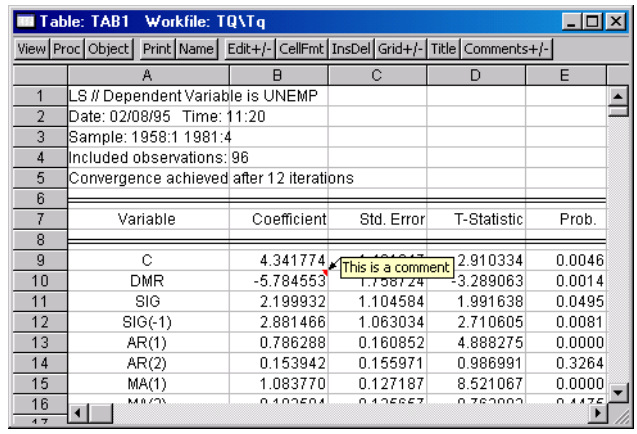


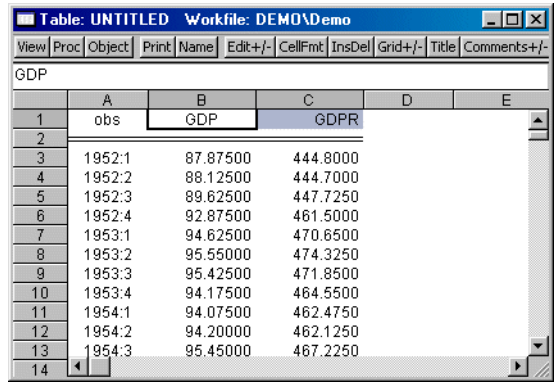
Table: TAB1 Workfile: TQ\Tq
View Proc Object Print Name Edit+/- CellFmt InsDel Grid+/- Title Comments+/-
LS // Dependent Variable is UNEMP
Date: 02/08/95 Time: 11:20
Sample: 1958:1 1981:4
Included observations: 96
Convergence achieved after 12 iterations
Variable Coefficient Std. Error T-Statistic Prob.
C 4.341774 2.910334 0.0046
DMR -5.784553 -3.289063 0.0014
SIG 2.199932 1.104584 1.991638 0.0495
SIG(-1) 2.881466 1.063034 2.710605 0.0081
AR(1) 0.786288 0.160852 4.888275 0.0000
AR(2) 0.153942 0.155971 0.986991 0.3264
MA(1) 1.083770 0.127187 8.521067 0.0000
This is a comment

Use the **Comments + /-** button in the tool bar to toggle comment mode on and off. Note that the red triangle and comment text will not be exported or printed.

Cell Merging

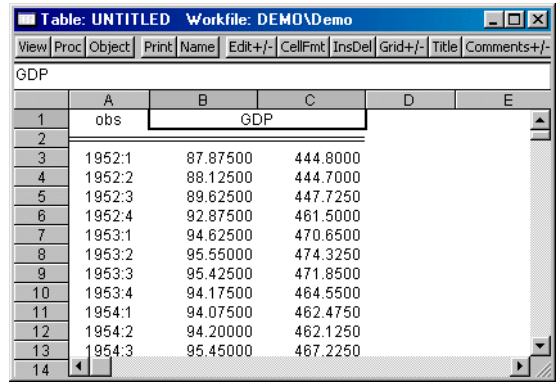
You may merge cells horizontally in a table object. When cells are merged, they are treated as a single cell for purposes of input, justification, and indentation. Merging cells is a useful tool in customizing the look of a table; it is, for example, an ideal way of centering text over multiple columns.

To merge several cells in a table row, simply select the individual cells you wish to merge, then right click and select **Merge Cell + /-**. EViews will merge the cells into a single cell. If the selected cells already contain any merged cells, the cells will be returned to their original state (*unmerged*).



Here, we begin by selecting the two cells B1 and C1. Note that B1 is the anchor cell, as indicated by the edit box surrounding the cell, and that B1 is center justified, while C1 is right justified. If we right mouse click and select **Merge Cell + /-**, the two cells will be merged, with the merged cell containing the contents and formatting of the anchor cell B1. If you wish C1 to be visible in the merged cell, you must alter the selection so that C1 is the anchor cell.

We see that the B1 and C1 cells are merged, as indicated by the large selection rectangle surrounding the merged cells. Editing the value of the merged cells will replace the value in the cell B1, but has no effect on hidden cells, in this case C1.



	A	B	C	D	E
1	obs	GDP			
2					
3	1952:1	87.87500	444.8000		
4	1952:2	88.12500	444.7000		
5	1952:3	89.62500	447.7250		
6	1952:4	92.87500	461.5000		
7	1953:1	94.62500	470.6500		
8	1953:2	95.55000	474.3250		
9	1953:3	95.42500	471.8500		
10	1953:4	94.17500	464.5500		
11	1954:1	94.07500	462.4750		
12	1954:2	94.20000	462.1250		
13	1954:3	95.45000	467.2250		
14					

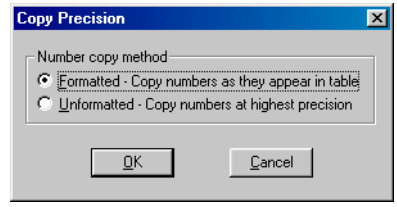
Bear in mind that the C1 cell has not been cleared; its contents have merely been hidden behind B1 in the merged cell.

If the merged cell is selected, toggling **Merge Cell +/-** will unmerge the cell so that cells are returned to their original form. The contents of C1 will once again be visible and may be modified using any of the table display formatting tools.

Copying Tables to the Clipboard

You may copy-and-paste a table to the Windows clipboard, from which you may paste the table contents into your favorite spreadsheet or word processing software.

Simply select the cells that you wish to copy and then choose **Edit/Copy** from the EViews main menu, or **Copy** from the right mouse button menu. The **Copy Precision** dialog box will open, providing you with the option of copying the numbers as they appear in the table, or at their highest internal precision.



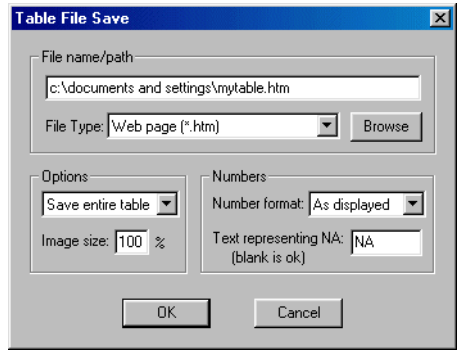
After you make a choice, EViews will place the table on the clipboard in Rich Text Format (RTF), allowing you to preserve the formatting information built into the table. Thus, if you copy-and-paste a table from EViews into Microsoft Word or another program which supports RTF, you will create a nicely formatted table containing your results.

To paste the clipboard contents into another application, switch to the destination application and select **Edit/Paste**. Note that some word processors provide the option of pasting the contents of the clipboard as unformatted files. If you wish to paste the table as unformatted text, you should select **Edit/Paste Special**.

Saving Tables to a File

EViews allows you to save your table objects in several file formats: Comma Separated Value (CSV), tab-delimited text (ASCII), Rich Text Format (RTF), or a Web page (HTML) file.

To save the table to disk, with the table window active or with table cells selected, right mouse click or select **Proc**, then select **Save table to disk...** to bring up the **Table File Save** dialog. The dialog displays default values from the global settings.



In the top portion of the dialog, you should provide the name of the file you wish to create. EViews will automatically append an extension of the proper type to the name (here, ".HTM" since we are saving a Web HTML file), and will prepend the default path if an explicit path is not provided.

Next, select the **File type**. You may select **Comma Separated Value**, **Tab Delimited Text-ASCII**, **Rich Text Format**, or **Web page**. The options section of the dialog allows you to save the entire table or only those cells that are currently selected, and, for HTML file output, to scale the table size.

You may also specify options for how numbers are to be treated when written. You may specify a **Number format** so that numbers are written **As displayed** in the table, or using **Full precision**. In addition, you may change the text used to write missing values.

Table Commands

EViews provides tools for performing extensive table customization from the command line or using programs. See "Table" (p. 187) in the *Command and Programming Reference* for additional details.

Text Objects

Some output views have no formatting and are simple displays of text information. Examples are representations of an equation and results from X-11 seasonal adjustment. If you freeze one of these views, you will create a text object.

You can also create a blank text object by selecting **Object/New Object.../Text** in the main EViews menu or by simply typing "`text`" in the command window. Text objects may be

used whenever you wish to capture textual data that does not contain any formatting information.

Part III. Basic Single Equation Analysis

The following chapters describe the EViews features for basic single equation analysis.

- Chapter 15, "Basic Regression", beginning on page 443 outlines the basics of ordinary least squares estimation in EViews.
- Chapter 16, "Additional Regression Methods", on page 461 discusses weighted least squares, two-stage least squares and nonlinear least square estimation techniques.
- Chapter 17, "Time Series Regression", on page 493 describes single equation regression techniques for the analysis of time series data: testing for serial correlation, estimation of ARMAX and ARIMAX models, using polynomial distributed lags, and unit root tests for nonstationary time series.
- Chapter 18, "Forecasting from an Equation", beginning on page 543 outlines the fundamentals of using EViews to forecast from estimated equations.
- Chapter 19, "Specification and Diagnostic Tests", beginning on page 569 describes specification testing in EViews.

The chapters describing advanced single equation techniques for autoregressive conditional heteroskedasticity, and discrete and limited dependent variable models are listed in Part IV "Advanced Single Equation Analysis".

Multiple equation estimation is described in the chapters listed in Part V "Multiple Equation Analysis".

Chapter 29, "Panel Estimation", beginning on page 901 describes estimation in panel structured workfiles.

Chapter 15. Basic Regression

Single equation regression is one of the most versatile and widely used statistical techniques. Here, we describe the use of basic regression techniques in EViews: specifying and estimating a regression model, performing simple diagnostic analysis, and using your estimation results in further analysis.

Subsequent chapters discuss testing and forecasting, as well as more advanced and specialized techniques such as weighted least squares, two-stage least squares (TSLS), nonlinear least squares, ARIMA/ARIMAX models, generalized method of moments (GMM), GARCH models, and qualitative and limited dependent variable models. These techniques and models all build upon the basic ideas presented in this chapter.

You will probably find it useful to own an econometrics textbook as a reference for the techniques discussed in this and subsequent documentation. Standard textbooks that we have found to be useful are listed below (in generally increasing order of difficulty):

- Pindyck and Rubinfeld (1991), *Econometric Models and Economic Forecasts*, 3*rd* edition.
- Johnston and DiNardo (1997), *Econometric Methods*, 4th Edition.
- Wooldridge (2000), *Introductory Econometrics: A Modern Approach*.
- Greene (1997), *Econometric Analysis*, 3rd Edition.
- Davidson and MacKinnon (1993), *Estimation and Inference in Econometrics*.

Where appropriate, we will also provide you with specialized references for specific topics.

Equation Objects

Single equation regression estimation in EViews is performed using the *equation object*. To create an equation object in EViews: select **Object/New Object.../Equation** or **Quick/Estimate Equation...** from the main menu, or simply type the keyword `equation` in the command window.

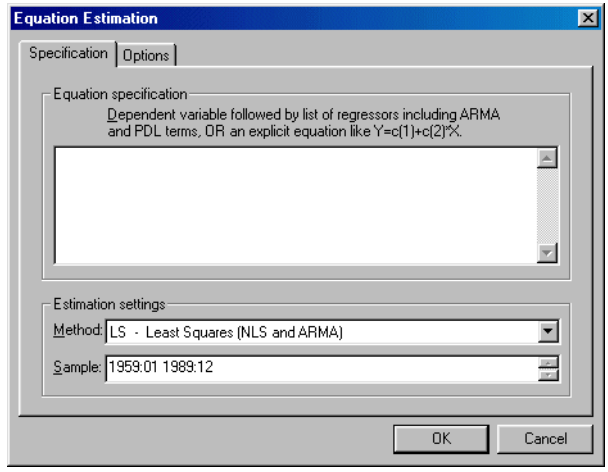
Next, you will specify your equation in the Equation Specification dialog box that appears, and select an estimation method. Below, we provide details on specifying equations in EViews. EViews will estimate the equation and display results in the equation window.

The estimation results are stored as part of the equation object so they can be accessed at any time. Simply open the object to display the summary results, or to access EViews tools for working with results from an equation object. For example, you can retrieve the sum-

of-squares from any equation, or you can use the estimated equation as part of a multi-equation model.

Specifying an Equation in EViews

When you create an equation object, a specification dialog box is displayed.



You need to specify three things in this dialog: the equation specification, the estimation method, and the sample to be used in estimation.

In the upper edit box, you can specify the equation: the dependent (left-hand side) and independent (right-hand side) variables and the functional form. There are two basic ways of specifying an equation: "by list" and "by formula" or "by expression". The list method is easier but may only be used with unrestricted linear specifications; the formula method is more general and must be used to specify nonlinear models or models with parametric restrictions.

Specifying an Equation by List

The simplest way to specify a linear equation is to provide a list of variables that you wish to use in the equation. First, include the name of the dependent variable or expression, followed by a list of explanatory variables. For example, to specify a linear consumption function, CS regressed on a constant and INC, type the following in the upper field of the **Equation Specification** dialog:

```
cs c inc
```

Note the presence of the series name C in the list of regressors. This is a built-in EViews series that is used to specify a constant in a regression. EViews does not automatically include a constant in a regression so you must explicitly list the constant (or its equivalent) as a regressor. The internal series C does not appear in your workfile, and you may not use it outside of specifying an equation. If you need a series of ones, you can generate a new series, or use the number 1 as an auto-series.

You may have noticed that there is a pre-defined object C in your workfile. This is the *default coefficient vector*—when you specify an equation by listing variable names, EViews

stores the estimated coefficients in this vector, in the order of appearance in the list. In the example above, the constant will be stored in C(1) and the coefficient on INC will be held in C(2).

Lagged series may be included in statistical operations using the same notation as in generating a new series with a formula—put the lag in parentheses after the name of the series. For example, the specification:

```
cs cs(-1) c inc
```

tells EViews to regress CS on its own lagged value, a constant, and INC. The coefficient for lagged CS will be placed in C(1), the coefficient for the constant is C(2), and the coefficient of INC is C(3).

You can include a consecutive range of lagged series by using the word "`to`" between the lags. For example:

```
cs c cs(-1 to -4) inc
```

regresses CS on a constant, CS(-1), CS(-2), CS(-3), CS(-4), and INC. If you don't include the first lag, it is taken to be zero. For example:

```
cs c inc(to -2) inc(-4)
```

regresses CS on a constant, INC, INC(-1), INC(-2), and INC(-4).

You may include auto-series in the list of variables. If the auto-series expressions contain spaces, they should be enclosed in parentheses. For example:

```
log(cs) c log(cs(-1)) ((inc+inc(-1)) / 2)
```

specifies a regression of the natural logarithm of CS on a constant, its own lagged value, and a two period moving average of INC.

Typing the list of series may be cumbersome, especially if you are working with many regressors. If you wish, EViews can create the specification list for you. First, highlight the dependent variable in the workfile window by single clicking on the entry. Next, CTRL-click on each of the explanatory variables to highlight them as well. When you are done selecting all of your variables, double click on any of the highlighted series, and select **Open/Equation...**, or right click and select **Open/as Equation...**. The **Equation Specification** dialog box should appear with the names entered in the specification field. The constant C is automatically included in this list; you must delete the C if you do not wish to include the constant.

Specifying an Equation by Formula

You will need to specify your equation using a formula when the list method is not general enough for your specification. Many, but not all, estimation methods allow you to specify your equation using a formula.

An equation formula in EViews is a mathematical expression involving regressors and coefficients. To specify an equation using a formula, simply enter the expression in the dialog in place of the list of variables. EViews will add an implicit additive disturbance to this equation and will estimate the parameters of the model using least squares.

When you specify an equation by list, EViews converts this into an equivalent equation formula. For example, the list,

```
log(cs) c log(cs(-1)) log(inc)
```

is interpreted by EViews as:

```
log(cs) = c(1) + c(2)*log(cs(-1)) + c(3)*log(inc)
```

Equations do not have to have a dependent variable followed by an equal sign and then an expression. The " = " sign can be anywhere in the formula, as in:

```
log(urate) - c(1)*dmr = c(2)
```

The residuals for this equation are given by:

$$\epsilon = \log(\text{urate}) - c(1)\text{dmr} - c(2)\,. \tag{15.1}$$

EViews will minimize the sum-of-squares of these residuals.

If you wish, you can specify an equation as a simple expression, without a dependent variable and an equal sign. If there is no equal sign, EViews assumes that the entire expression is the disturbance term. For example, if you specify an equation as:

```
c(1)*x + c(2)*y + 4*z
```

EViews will find the coefficient values that minimize the sum of squares of the given expression, in this case (C(1)*X + C(2)*Y + 4*Z). While EViews will estimate an expression of this type, since there is no dependent variable, some regression statistics (*e.g.* R-squared) are not reported and the equation cannot be used for forecasting. This restriction also holds for any equation that includes coefficients to the left of the equal sign. For example, if you specify:

```
x + c(1)*y = c(2)*z
```

EViews finds the values of C(1) and C(2) that minimize the sum of squares of (X + C(1)*Y–C(2)*Z). The estimated coefficients will be identical to those from an equation specified using:

```
x = -c(1)*y + c(2)*z
```

but some regression statistics are not reported.

The two most common motivations for specifying your equation by formula are to estimate restricted and nonlinear models. For example, suppose that you wish to constrain the coefficients on the lags on the variable X to sum to one. Solving out for the coefficient restriction leads to the following linear model with parameter restrictions:

```
y = c(1) + c(2)*x + c(3)*x(-1) + c(4)*x(-2) +(1-c(2)-c(3)-
    c(4))*x(-3)
```

To estimate a nonlinear model, simply enter the nonlinear formula. EViews will automatically detect the nonlinearity and estimate the model using nonlinear least squares. For details, see "Nonlinear Least Squares" on page 480.

One benefit to specifying an equation by formula is that you can elect to use a different coefficient vector. To create a new coefficient vector, choose **Object/New Object...** and select **Matrix-Vector-Coef** from the main menu, type in a name for the coefficient vector, and click **OK**. In the New Matrix dialog box that appears, select **Coefficient Vector** and specify how many rows there should be in the vector. The object will be listed in the workfile directory with the coefficient vector icon (the little β).

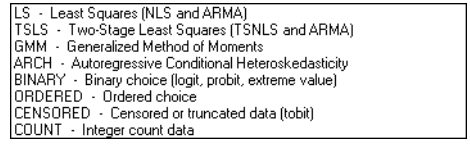
You may then use this coefficient vector in your specification. For example, suppose you created coefficient vectors A and BETA, each with a single row. Then you can specify your equation using the new coefficients in place of C:

```
log(cs) = a(1) + beta(1)*log(cs(-1))
```

Estimating an Equation in EViews

Estimation Methods

Having specified your equation, you now need to choose an estimation method. Click on the **Method:** entry in the dialog and you will see a drop-down menu listing estimation methods.


LS - Least Squares (NLS and ARMA)
TSLS - Two-Stage Least Squares (TSNLS and ARMA)
GMM - Generalized Method of Moments
ARCH - Autoregressive Conditional Heteroskedasticity
BINARY - Binary choice (logit, probit, extreme value)
ORDERED - Ordered choice
CENSORED - Censored or truncated data (tobit)
COUNT - Integer count data

Standard, single-equation regression is performed using least squares. The other methods are described in subsequent chapters.

Equations estimated by ordinary least squares and two-stage least squares, equations with AR terms, GMM, and ARCH equations may be specified with a formula. Expression equations are not allowed with binary, ordered, censored, and count models, or in equations with MA terms.

Estimation Sample

You should also specify the sample to be used in estimation. EViews will fill out the dialog with the current workfile sample, but you can change the sample for purposes of estimation by entering your sample string or object in the edit box (see "Samples" on page 95 for details). Changing the estimation sample does not affect the current workfile sample.

If any of the series used in estimation contain missing data, EViews will temporarily adjust the estimation sample of observations to exclude those observations (listwise exclusion). EViews notifies you that it has adjusted the sample by reporting the actual sample used in the estimation results:

```
Dependent Variable: Y
Method: Least Squares
Date: 08/19/97   Time: 10:24
Sample(adjusted): 1959:01 1989:12
Included observations: 340
Excluded observations: 32 after adjusting endpoints
```

Here we see the top of an equation output view. EViews reports that it has adjusted the sample. Out of the 372 observations in the period 1959M01–1989M12, EViews uses the 340 observations with observations for all of the relevant variables.

You should be aware that if you include lagged variables in a regression, the degree of sample adjustment will differ depending on whether data for the pre-sample period are available or not. For example, suppose you have nonmissing data for the two series M1 and IP over the period 1959:01–1989:12 and specify the regression as:

```
m1 c ip ip(-1) ip(-2) ip(-3)
```

If you set the estimation sample to the period 1959:01–1989:12, EViews adjusts the sample to:

```
Dependent Variable: M1
Method: Least Squares
Date: 08/19/97   Time: 10:49
Sample: 1960:01 1989:12
Included observations: 360
```

since data for IP(–3) are not available until 1959M04. However, if you set the estimation sample to the period 1960M01–1989M12, EViews will not make any adjustment to the sample since all values of IP(-3) are available during the estimation sample.

Some operations, most notably estimation with MA terms and ARCH, do not allow missing observations in the middle of the sample. When executing these procedures, an error message is displayed and execution is halted if an NA is encountered in the middle of the sample. EViews handles missing data at the very start or the very end of the sample range by adjusting the sample endpoints and proceeding with the estimation procedure.

Estimation Options

EViews provides a number of estimation options. These options allow you to weight the estimating equation, to compute heteroskedasticity and auto-correlation robust covariances, and to control various features of your estimation algorithm. These options are discussed in detail in "Estimation Options" on page 482.

Equation Output

When you click **OK** in the Equation Specification dialog, EViews displays the equation window displaying the estimation output view:

Dependent Variable: LOG(M1)
Method: Least Squares
Date: 08/18/97 Time: 14:02
Sample: 1959:01 1989:12
Included observations: 372

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-1.699912	0.164954	-10.30539	0.0000
LOG(IP)	1.765866	0.043546	40.55199	0.0000
TB3	-0.011895	0.004628	-2.570016	0.0106

R-squared	0.886416	Mean dependent var	5.663717
Adjusted R-squared	0.885800	S.D. dependent var	0.553903
S.E. of regression	0.187183	Akaike info criterion	-0.505429
Sum squared resid	12.92882	Schwarz criterion	-0.473825
Log likelihood	97.00980	F-statistic	1439.848
Durbin-Watson stat	0.008687	Prob(F-statistic)	0.000000

Using matrix notation, the standard regression may be written as:

$$y = X\beta + \epsilon \tag{15.2}$$

where y is a T-dimensional vector containing observations on the dependent variable, X is a $T \times k$ matrix of independent variables, β is a k-vector of coefficients, and ϵ is a T-vector of disturbances. T is the number of observations and k is the number of right-hand side regressors.

In the output above, y is log(M1), X consists of three variables C, log(IP), and TB3, where $T = 372$ and $k = 3$.

Coefficient Results

Regression Coefficients

The column labeled "Coefficient" depicts the estimated coefficients. The least squares regression coefficients b are computed by the standard OLS formula:

$$b = (X'X)^{-1}X'y \tag{15.3}$$

If your equation is specified by list, the coefficients will be labeled in the "Variable" column with the name of the corresponding regressor; if your equation is specified by formula, EViews lists the actual coefficients, C(1), C(2), *etc*.

For the simple linear models considered here, the coefficient measures the marginal contribution of the independent variable to the dependent variable, holding all other variables fixed. If present, the coefficient of the C is the constant or intercept in the regression—it is the base level of the prediction when all of the other independent variables are zero. The other coefficients are interpreted as the slope of the relation between the corresponding independent variable and the dependent variable, assuming all other variables do not change.

Standard Errors

The "Std. Error" column reports the estimated standard errors of the coefficient estimates. The standard errors measure the statistical reliability of the coefficient estimates—the larger the standard errors, the more statistical noise in the estimates. If the errors are normally distributed, there are about 2 chances in 3 that the true regression coefficient lies within one standard error of the reported coefficient, and 95 chances out of 100 that it lies within two standard errors.

The covariance matrix of the estimated coefficients is computed as:

$$\text{var}(b) = s^2(X'X)^{-1}; \quad s^2 = \hat{\epsilon}'\hat{\epsilon}/(T-k); \quad \hat{\epsilon} = y - Xb \tag{15.4}$$

where $\hat{\epsilon}$ is the residual. The standard errors of the estimated coefficients are the square roots of the diagonal elements of the coefficient covariance matrix. You can view the whole covariance matrix by choosing **View/Covariance Matrix**.

t-Statistics

The t-statistic, which is computed as the ratio of an estimated coefficient to its standard error, is used to test the hypothesis that a coefficient is equal to zero. To interpret the t-statistic, you should examine the probability of observing the t-statistic given that the coefficient is equal to zero. This probability computation is described below.

In cases where normality can only hold asymptotically, EViews will report a z-statistic instead of a t-statistic.

Probability

The last column of the output shows the probability of drawing a t-statistic (or a z-statistic) as extreme as the one actually observed, under the assumption that the errors are normally distributed, or that the estimated coefficients are asymptotically normally distributed.

This probability is also known as the *p-value* or the *marginal significance level*. Given a p-value, you can tell at a glance if you reject or accept the hypothesis that the true coefficient

is zero against a two-sided alternative that it differs from zero. For example, if you are performing the test at the 5% significance level, a p-value lower than 0.05 is taken as evidence to reject the null hypothesis of a zero coefficient. If you want to conduct a one-sided test, the appropriate probability is one-half that reported by EViews.

For the above example output, the hypothesis that the coefficient on TB3 is zero is rejected at the 5% significance level but not at the 1% level. However, if theory suggests that the coefficient on TB3 cannot be positive, then a one-sided test will reject the zero null hypothesis at the 1% level.

The p-values are computed from a t-distribution with $T-k$ degrees of freedom.

Summary Statistics

R-squared

The R-squared (R^2) statistic measures the success of the regression in predicting the values of the dependent variable within the sample. In standard settings, R^2 may be interpreted as the fraction of the variance of the dependent variable explained by the independent variables. The statistic will equal one if the regression fits perfectly, and zero if it fits no better than the simple mean of the dependent variable. It can be negative for a number of reasons. For example, if the regression does not have an intercept or constant, if the regression contains coefficient restrictions, or if the estimation method is two-stage least squares or ARCH.

EViews computes the (centered) R^2 as:

$$R^2 = 1 - \frac{\hat{\epsilon}'\hat{\epsilon}}{(y-\bar{y})'(y-\bar{y})}; \quad \bar{y} = \sum_{t=1}^{T} y_t/T \tag{15.5}$$

where $\bar{y}$ is the mean of the dependent (left-hand) variable.

Adjusted R-squared

One problem with using R^2 as a measure of goodness of fit is that the R^2 will never decrease as you add more regressors. In the extreme case, you can always obtain an R^2 of one if you include as many independent regressors as there are sample observations.

The adjusted R^2, commonly denoted as $\overline{R}^2$, penalizes the R^2 for the addition of regressors which do not contribute to the explanatory power of the model. The adjusted R^2 is computed as:

$$\overline{R}^2 = 1 - (1-R^2)\frac{T-1}{T-k} \tag{15.6}$$

The $\overline{R}^2$ is never larger than the R^2, can decrease as you add regressors, and for poorly fitting models, may be negative.

Standard Error of the Regression (S.E. of regression)

The standard error of the regression is a summary measure based on the estimated variance of the residuals. The standard error of the regression is computed as:

$$s = \sqrt{\frac{\hat{\epsilon}'\hat{\epsilon}}{(T-k)}} \tag{15.7}$$

Sum-of-Squared Residuals

The sum-of-squared residuals can be used in a variety of statistical calculations, and is presented separately for your convenience:

$$\hat{\epsilon}'\hat{\epsilon} = \sum_{t=1}^{T} (y_i - X_i'b)^2 \tag{15.8}$$

Log Likelihood

EViews reports the value of the log likelihood function (assuming normally distributed errors) evaluated at the estimated values of the coefficients. Likelihood ratio tests may be conducted by looking at the difference between the log likelihood values of the restricted and unrestricted versions of an equation.

The log likelihood is computed as:

$$l = -\frac{T}{2}(1 + \log(2\pi) + \log(\hat{\epsilon}'\hat{\epsilon}/T)) \tag{15.9}$$

When comparing EViews output to that reported from other sources, note that EViews does not ignore constant terms.

Durbin-Watson Statistic

The Durbin-Watson statistic measures the serial correlation in the residuals. The statistic is computed as

$$\text{DW} = \sum_{t=2}^{T} (\hat{\epsilon}_t - \hat{\epsilon}_{t-1})^2 / \sum_{t=1}^{T} \hat{\epsilon}_t^2 \tag{15.10}$$

See Johnston and DiNardo (1997, Table D.5) for a table of the significance points of the distribution of the Durbin-Watson statistic.

As a rule of thumb, if the DW is less than 2, there is evidence of positive serial correlation. The DW statistic in our output is very close to one, indicating the presence of serial correlation in the residuals. See "Serial Correlation Theory" beginning on page 493 for a more extensive discussion of the Durbin-Watson statistic and the consequences of serially correlated residuals.

There are better tests for serial correlation. In "Testing for Serial Correlation" on page 494, we discuss the Q-statistic, and the Breusch-Godfrey LM test, both of which provide a more general testing framework than the Durbin-Watson test.

Mean and Standard Deviation (S.D.) of the Dependent Variable

The mean and standard deviation of y are computed using the standard formulae:

$$\bar{y} = \sum_{t=1}^{T} y_t / T; \qquad s_y = \sqrt{\sum_{t=1}^{T} (y_t - \bar{y})^2 / (T-1)} \tag{15.11}$$

Akaike Information Criterion

The Akaike Information Criterion (AIC) is computed as:

$$\text{AIC} = -2l/T + 2k/T \tag{15.12}$$

where l is the log likelihood (given by Equation (15.9) on page 452).

The AIC is often used in model selection for non-nested alternatives—smaller values of the AIC are preferred. For example, you can choose the length of a lag distribution by choosing the specification with the lowest value of the AIC. See Appendix E, "Information Criteria", on page 971, for additional discussion.

Schwarz Criterion

The Schwarz Criterion (SC) is an alternative to the AIC that imposes a larger penalty for additional coefficients:

$$\text{SC} = -2l/T + (k \log T)/T \tag{15.13}$$

F-Statistic

The F-statistic reported in the regression output is from a test of the hypothesis that *all* of the slope coefficients (excluding the constant, or intercept) in a regression are zero. For ordinary least squares models, the F-statistic is computed as:

$$F = \frac{R^2/(k-1)}{(1-R^2)/(T-k)} \tag{15.14}$$

Under the null hypothesis with normally distributed errors, this statistic has an F-distribution with $k-1$ numerator degrees of freedom and $T-k$ denominator degrees of freedom.

The p-value given just below the F-statistic, denoted **Prob(F-statistic)**, is the marginal significance level of the F-test. If the p-value is less than the significance level you are testing, say 0.05, you reject the null hypothesis that all slope coefficients are equal to zero. For the example above, the p-value is essentially zero, so we reject the null hypothesis that all of

the regression coefficients are zero. Note that the F-test is a joint test so that even if all the t-statistics are insignificant, the F-statistic can be highly significant.

Working With Equation Statistics

The regression statistics reported in the estimation output view are stored with the equation and are accessible through special "@-functions". You can retrieve any of these statistics for further analysis by using these functions in genr, scalar, or matrix expressions. If a particular statistic is not computed for a given estimation method, the function will return an NA.

There are two kinds of "@-functions": those that return a scalar value, and those that return matrices or vectors.

Keywords that return scalar values

@aic	Akaike information criterion
@coefcov(i,j)	covariance of coefficient estimates i and j
@coefs(i)	i-th coefficient value
@dw	Durbin-Watson statistic
@f	F-statistic
@hq	Hannan-Quinn information criterion
@jstat	J-statistic — value of the GMM objective function (for GMM)
@logl	value of the log likelihood function
@meandep	mean of the dependent variable
@ncoef	number of estimated coefficients
@r2	R-squared statistic
@rbar2	adjusted R-squared statistic
@regobs	number of observations in regression
@schwarz	Schwarz information criterion
@sddep	standard deviation of the dependent variable
@se	standard error of the regression
@ssr	sum of squared residuals
@stderrs(i)	standard error for coefficient i
@tstats(i)	t-statistic value for coefficient i
c(i)	i-th element of default coefficient vector for equation (if applicable)

Keywords that return vector or matrix objects

@coefcov	matrix containing the coefficient covariance matrix
@coefs	vector of coefficient values
@stderrs	vector of standard errors for the coefficients
@tstats	vector of *t*-statistic values for coefficients

See also "Equation" (p. 157) in the *Command and Programming Reference*.

Functions that return a vector or matrix object should be assigned to the corresponding object type. For example, you should assign the results from `@tstats` to a vector:

```
vector tstats = eq1.@tstats
```

and the covariance matrix to a matrix:

```
matrix mycov = eq1.@cov
```

You can also access individual elements of these statistics:

```
scalar pvalue = 1-@cnorm(@abs(eq1.@tstats(4)))
scalar var1 = eq1.@covariance(1,1)
```

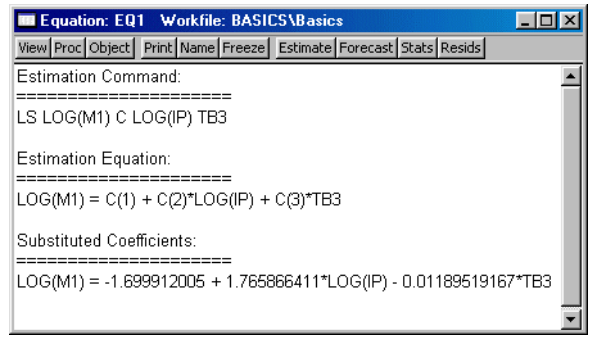
For documentation on using vectors and matrices in EViews, see Chapter 3, "Matrix Language", on page 23 of the *Command and Programming Reference*.

Working with Equations

Views of an Equation

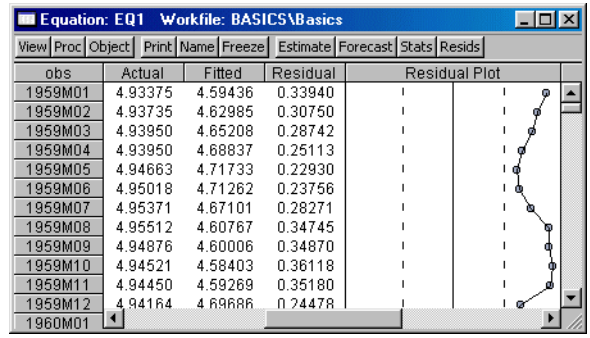
- **Representations**. Displays the equation in three forms: EViews command form, as an algebraic equation with symbolic coefficients, and as an equation with the estimated values of the coefficients.

  You can cut-and-paste from the representations view into any application that supports the Windows clipboard.


Equation: EQ1 Workfile: BASICS\Basics
View Proc Object Print Name Freeze Estimate Forecast Stats Resids
Estimation Command:
LS LOG(M1) C LOG(IP) TB3
Estimation Equation:
LOG(M1) = C(1) + C(2)*LOG(IP) + C(3)*TB3
Substituted Coefficients:
LOG(M1) = -1.699912005 + 1.765866411*LOG(IP) - 0.01189519167*TB3

- **Estimation Output**. Displays the equation output results described above.
- **Actual, Fitted, Residual**. These views display the

actual and fitted values of the dependent variable and the residuals from the regression in tabular and graphical form. **Actual, Fitted, Residual Table** displays these values in table form.



Note that the actual value is always the sum of the fitted value and the residual. **Actual, Fitted, Residual Graph** displays a standard EViews graph of the actual values, fitted values, and residuals. **Residual Graph** plots only the residuals, while the **Standardized Residual Graph** plots the residuals divided by the estimated residual standard deviation.

- **ARMA structure...**. Provides views which describe the estimated ARMA structure of your residuals. Details on these views are provided in "ARMA Structure" on page 512.

- **Gradients and Derivatives...**. Provides views which describe the gradients of the objective function and the information about the computation of any derivatives of the regression function. Details on these views are provided in Appendix D, "Gradients and Derivatives", on page 963.

- **Covariance Matrix**. Displays the covariance matrix of the coefficient estimates as a spreadsheet view. To save this covariance matrix as a matrix object, use the `@cov` function.

- **Coefficient Tests, Residual Tests, and Stability Tests**. These are views for specification and diagnostic tests and are described in detail in Chapter 19, "Specification and Diagnostic Tests", beginning on page 569.

Procedures of an Equation

- **Specify/Estimate...**. Brings up the **Equation Specification** dialog box so that you can modify your specification. You can edit the equation specification, or change the estimation method or estimation sample.

- **Forecast...**. Forecasts or fits values using the estimated equation. Forecasting using equations is discussed in Chapter 18, "Forecasting from an Equation", on page 543.

- **Make Residual Series...**. Saves the residuals from the regression as a series in the workfile. Depending on the estimation method, you may choose from three types of

residuals: ordinary, standardized, and generalized. For ordinary least squares, only the ordinary residuals may be saved.

- **Make Regressor Group.** Creates an untitled group comprised of all the variables used in the equation (with the exception of the constant).
- **Make Gradient Group**. Creates a group containing the gradients of the objective function with respect to the coefficients of the model.
- **Make Derivative Group.** Creates a group containing the derivatives of the regression function with respect to the coefficients in the regression function.
- **Make Model**. Creates an untitled model containing a link to the estimated equation. This model can be solved in the usual manner. See Chapter 26, "Models", on page 777 for information on how to use models for forecasting and simulations.
- **Update Coefs from Equation.** Places the estimated coefficients of the equation in the coefficient vector. You can use this procedure to initialize starting values for various estimation procedures.

Residuals from an Equation

The residuals from the default equation are stored in a series object called RESID. RESID may be used directly as if it were a regular series, except in estimation.

RESID will be overwritten whenever you estimate an equation and will contain the residuals from the latest estimated equation. To save the residuals from a particular equation for later analysis, you should save them in a different series so they are not overwritten by the next estimation command. For example, you can copy the residuals into a regular EViews series called RES1 by the command:

```
series res1 = resid
```

Even if you have already overwritten the RESID series, you can always create the desired series using EViews' built-in procedures if you still have the equation object. If your equation is named EQ1, open the equation window and select **Proc/Make Residual Series,** or enter:

```
eq1.makeresid res1
```

to create the desired series.

Regression Statistics

You may refer to various regression statistics through the @-functions described above. For example, to generate a new series equal to FIT plus twice the standard error from the last regression, you can use the command:

```
series plus = fit + 2*eq1.@se
```

To get the *t*-statistic for the second coefficient from equation EQ1, you could specify

```
eq1.@tstats(2)
```

To store the coefficient covariance matrix from EQ1 as a named symmetric matrix, you can use the command:

```
sym ccov1 = eq1.@cov
```

See "Keywords that return scalar values" on page 454 for additional details.

Storing and Retrieving an Equation

As with other objects, equations may be stored to disk in data bank or database files. You can also fetch equations from these files.

Equations may also be copied-and-pasted to, or from, workfiles or databases.

EViews even allows you to access equations directly from your databases or another workfile. You can estimate an equation, store it in a database, and then use it to forecast in several workfiles.

See Chapter 4, "Object Basics", beginning on page 73 and Chapter 10, "EViews Databases", beginning on page 261 for additional information about objects, databases, and object containers.

Using Estimated Coefficients

The coefficients of an equation are listed in the representations view. By default, EViews will use the C coefficient vector when you specify an equation, but you may explicitly use other coefficient vectors in defining your equation.

These stored coefficients may be used as scalars in generating data. While there are easier ways of generating fitted values (see "Forecasting from an Equation" on page 543), for purposes of illustration, note that we can use the coefficients to form the fitted values from an equation. The command:

```
series cshat = eq1.c(1) + eq1.c(2)*gdp
```

forms the fitted value of CS, CSHAT, from the OLS regression coefficients and the independent variables from the equation object EQ1.

Note that while EViews will accept a series generating equation which does not explicitly refer to a named equation:

```
series cshat = c(1) + c(2)*gdp
```

and will use the existing values in the C coefficient vector, we strongly recommend that you always use named equations to identify the appropriate coefficients. In general, C will contain the correct coefficient values only immediately following estimation or a coefficient update. Using a named equation, or selecting **Proc/Update coefs from equation,** guarantees that you are using the correct coefficient values.

An alternative to referring to the coefficient vector is to reference the `@coefs` elements of your equation (see page 454). For example, the examples above may be written as:

```
series cshat=eq1.@coefs(1)+eq1.@coefs(2)*gdp
```

EViews assigns an index to each coefficient in the order that it appears in the representations view. Thus, if you estimate the equation:

```
equation eq01.ls y=c(10)+b(5)*y(-1)+a(7)*inc
```

where B and A are also coefficient vectors, then:

- eq01.@coefs(1) contains C(10)
- eq01.@coefs(2) contains B(5)
- eq01.@coefs(3) contains A(7)

This method should prove useful in matching coefficients to standard errors derived from the `@stderrs` elements of the equation (see Appendix A, "Object, View and Procedure Reference", on page 153 of the *Command and Programming Reference*). The `@coefs` elements allow you to refer to both the coefficients and the standard errors using a common index.

If you have used an alternative named coefficient vector in specifying your equation, you can also access the coefficient vector directly. For example, if you have used a coefficient vector named BETA, you can generate the fitted values by issuing the commands:

```
equation eq02.ls cs=beta(1)+beta(2)*gdp
series cshat=beta(1)+beta(2)*gdp
```

where BETA is a coefficient vector. Again, however, we recommend that you use the `@coefs` elements to refer to the coefficients of EQ02. Alternatively, you can update the coefficients in BETA prior to use by selecting **Proc/Update coefs from equation** from the equation window. Note that EViews does not allow you to refer to the named equation coefficients EQ02.BETA(1) and EQ02.BETA(2). You must instead use the expressions, EQ02.@COEFS(1) and EQ02.@COEFS(2).

Estimation Problems

Exact Collinearity

If the regressors are very highly collinear, EViews may encounter difficulty in computing the regression estimates. In such cases, EViews will issue an error message "Near singular matrix." When you get this error message, you should check to see whether the regressors are *exactly* collinear. The regressors are exactly collinear if one regressor can be written as a linear combination of the other regressors. Under exact collinearity, the regressor matrix X does not have full column rank and the OLS estimator cannot be computed.

You should watch out for exact collinearity when you are using dummy variables in your regression. A set of mutually exclusive dummy variables and the constant term are exactly collinear. For example, suppose you have quarterly data and you try to run a regression with the specification:

```
y c x @seas(1) @seas(2) @seas(3) @seas(4)
```

EViews will return a "Near singular matrix" error message since the constant and the four quarterly dummy variables are exactly collinear through the relation:

```
c = @seas(1) + @seas(2) + @seas(3) + @seas(4)
```

In this case, simply drop either the constant term or one of the dummy variables.

The textbooks listed above provide extensive discussion of the issue of collinearity.

Chapter 16. Additional Regression Methods

This first portion of this chapter describes special terms that may be used in estimation to estimate models with Polynomial Distributed Lags (PDLs) or dummy variables.

In addition, we describe weighted least squares, heteroskedasticity and autocorrelation consistent covariance estimation, two-stage least squares (TSLS), nonlinear least squares, and generalized method of moments (GMM). Note that most of these methods are also available in systems of equations; see Chapter 23, "System Estimation", on page 696.

Parts of this chapter refer to estimation of models which have autoregressive (AR) and moving average (MA) error terms. These concepts are discussed in greater depth in Chapter 17, "Time Series Regression", on page 493.

Special Equation Terms

EViews provides you with special terms that may be used to specify and estimate equations with PDLs, dummy variables, or ARMA errors. We begin with a discussion of PDLs and dummy variables, and defer the discussion of ARMA estimation to "Time Series Regression" on page 493.

Polynomial Distributed Lags (PDLs)

A distributed lag is a relation of the type:

$$y_t = w_t\delta + \beta_0 x_t + \beta_1 x_{t-1} + \ldots + \beta_k x_{t-k} + \epsilon_t \tag{16.1}$$

The coefficients β describe the lag in the effect of x on y. In many cases, the coefficients can be estimated directly using this specification. In other cases, the high collinearity of current and lagged values of x will defeat direct estimation.

You can reduce the number of parameters to be estimated by using polynomial distributed lags (PDLs) to impose a smoothness condition on the lag coefficients. Smoothness is expressed as requiring that the coefficients lie on a polynomial of relatively low degree. A polynomial distributed lag model with order p restricts the β coefficients to lie on a p-th order polynomial of the form,

$$\beta_j = \gamma_1 + \gamma_2(j-\bar{c}) + \gamma_3(j-\bar{c})^2 + \ldots + \gamma_{p+1}(j-\bar{c})^p \tag{16.2}$$

for $j = 1, 2, \ldots, k$, where $\bar{c}$ is a pre-specified constant given by:

$$\bar{c} = \begin{cases} (k)/2 & \text{if } p \text{ is even} \\ (k-1)/2 & \text{if } p \text{ is odd} \end{cases} \tag{16.3}$$

The PDL is sometimes referred to as an Almon lag. The constant $\bar{c}$ is included only to avoid numerical problems that can arise from collinearity and does not affect the estimates of β.

This specification allows you to estimate a model with k lags of x using only p parameters (if you choose $p > k$, EViews will return a "Near Singular Matrix" error).

If you specify a PDL, EViews substitutes Equation (16.2) into (16.1), yielding,

$$y_t = \alpha + \gamma_1 z_1 + \gamma_2 z_2 + \ldots + \gamma_{p+1} z_{p+1} + \epsilon_t \tag{16.4}$$

where:

$$\begin{aligned} z_1 &= x_t + x_{t-1} + \ldots + x_{t-k} \\ z_2 &= -\bar{c}x_t + (1-\bar{c})x_{t-1} + \ldots + (k-\bar{c})x_{t-k} \\ &\ldots \\ z_{p+1} &= (-\bar{c})^p x_t + (1-\bar{c})^p x_{t-1} + \ldots + (k-\bar{c})^p x_{t-k} \end{aligned} \tag{16.5}$$

Once we estimate γ from Equation (16.4), we can recover the parameters of interest β, and their standard errors using the relationship described in Equation (16.2). This procedure is straightforward since β is a linear transformation of γ.

The specification of a polynomial distributed lag has three elements: the length of the lag k, the degree of the polynomial (the highest power in the polynomial) p, and the constraints that you want to apply. A near end constraint restricts the one-period lead effect of x on y to be zero:

$$\beta_{-1} = \gamma_1 + \gamma_2(-1-\bar{c}) + \ldots + \gamma_{p+1}(-1-\bar{c})^p = 0\,. \tag{16.6}$$

A far end constraint restricts the effect of x on y to die off beyond the number of specified lags:

$$\beta_{k+1} = \gamma_1 + \gamma_2(k+1-\bar{c}) + \ldots + \gamma_{p+1}(k+1-\bar{c})^p = 0\,. \tag{16.7}$$

If you restrict either the near or far end of the lag, the number of γ parameters estimated is reduced by one to account for the restriction; if you restrict both the near and far end of the lag, the number of γ parameters is reduced by two.

By default, EViews does not impose constraints.

How to Estimate Models Containing PDLs

You specify a polynomial distributed lag by the `pdl` term, with the following information in parentheses, each separated by a comma in this order:

- The name of the series.
- The lag length (the number of lagged values of the series to be included).

- The degree of the polynomial.
- A numerical code to constrain the lag polynomial (optional):

1	constrain the near end of the lag to zero.
2	constrain the far end.
3	constrain both ends.

You may omit the constraint code if you do not want to constrain the lag polynomial. Any number of `pdl` terms may be included in an equation. Each one tells EViews to fit distributed lag coefficients to the series and to constrain the coefficients to lie on a polynomial.

For example, the commands:

```
ls sales c pdl(orders,8,3)
```

fits SALES to a constant, and a distributed lag of current and eight lags of ORDERS, where the lag coefficients of ORDERS lie on a third degree polynomial with no endpoint constraints. Similarly:

```
ls div c pdl(rev,12,4,2)
```

fits DIV to a distributed lag of current and 12 lags of REV, where the coefficients of REV lie on a 4th degree polynomial with a constraint at the far end.

The `pdl` specification may also be used in two-stage least squares. If the series in the `pdl` is exogenous, you should include the PDL of the series in the instruments as well. For this purpose, you may specify `pdl(*)` as an instrument; all `pdl` variables will be used as instruments. For example, if you specify the TSLS equation as,

```
sales c inc pdl(orders(-1),12,4)
```

with instruments:

```
fed fed(-1) pdl(*)
```

the distributed lag of ORDERS will be used as instruments together with FED and FED(–1).

Polynomial distributed lags cannot be used in nonlinear specifications.

Example

The distributed lag model of industrial production (IP) on money (M1) yields the following results:

Dependent Variable: IP
Method: Least Squares
Date: 08/15/97 Time: 17:09
Sample(adjusted): 1960:01 1989:12
Included observations: 360 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	40.67568	0.823866	49.37171	0.0000
M1	0.129699	0.214574	0.604449	0.5459
M1(-1)	-0.045962	0.376907	-0.121944	0.9030
M1(-2)	0.033183	0.397099	0.083563	0.9335
M1(-3)	0.010621	0.405861	0.026169	0.9791
M1(-4)	0.031425	0.418805	0.075035	0.9402
M1(-5)	-0.048847	0.431728	-0.113143	0.9100
M1(-6)	0.053880	0.440753	0.122245	0.9028
M1(-7)	-0.015240	0.436123	-0.034944	0.9721
M1(-8)	-0.024902	0.423546	-0.058795	0.9531
M1(-9)	-0.028048	0.413540	-0.067825	0.9460
M1(-10)	0.030806	0.407523	0.075593	0.9398
M1(-11)	0.018509	0.389133	0.047564	0.9621
M1(-12)	-0.057373	0.228826	-0.250728	0.8022

R-squared	0.852398	Mean dependent var	71.72679
Adjusted R-squared	0.846852	S.D. dependent var	19.53063
S.E. of regression	7.643137	Akaike info criterion	6.943606
Sum squared resid	20212.47	Schwarz criterion	7.094732
Log likelihood	-1235.849	F-statistic	153.7030
Durbin-Watson stat	0.008255	Prob(F-statistic)	0.000000

Taken individually, none of the coefficients on lagged M1 are statistically different from zero. Yet the regression as a whole has a reasonable R^2 with a very significant F-statistic (though with a very low Durbin-Watson statistic). This is a typical symptom of high collinearity among the regressors and suggests fitting a polynomial distributed lag model.

To estimate a fifth-degree polynomial distributed lag model with no constraints, set the sample using the command,

```
smpl 1959:01 1989:12
```

then estimate the equation specification:

```
ip c pdl(m1,12,5)
```

using a command with the specification, or by entering the specification in the **Equation Estimation** dialog.

The following result is reported at the top of the equation window:

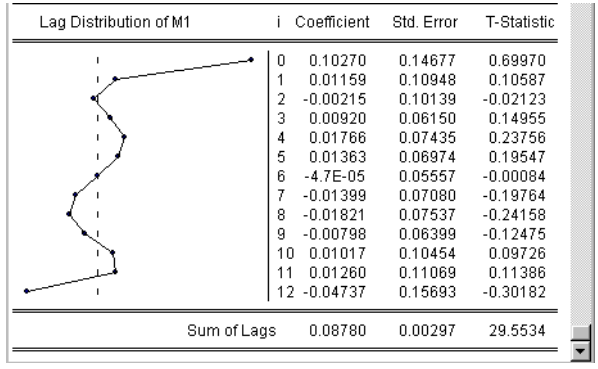
Dependent Variable: IP
Method: Least Squares
Date: 08/15/97 Time: 17:53
Sample(adjusted): 1960:01 1989:12
Included observations: 360 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	40.67311	0.815195	49.89374	0.0000
PDL01	-4.66E-05	0.055566	-0.000839	0.9993
PDL02	-0.015625	0.062884	-0.248479	0.8039
PDL03	-0.000160	0.013909	-0.011485	0.9908
PDL04	0.001862	0.007700	0.241788	0.8091
PDL05	2.58E-05	0.000408	0.063211	0.9496
PDL06	-4.93E-05	0.000180	-0.273611	0.7845

R-squared	0.852371	Mean dependent var	71.72679
Adjusted R-squared	0.849862	S.D. dependent var	19.53063
S.E. of regression	7.567664	Akaike info criterion	6.904899
Sum squared resid	20216.15	Schwarz criterion	6.980462
Log likelihood	-1235.882	F-statistic	339.6882
Durbin-Watson stat	0.008026	Prob(F-statistic)	0.000000

This portion of the view reports the estimated coefficients γ of the polynomial in Equation (16.2) on page 461. The terms PDL01, PDL02, PDL03, …, correspond to z_1, z_2, … in Equation (16.4).

The implied coefficients of interest β_j in equation (1) are reported at the bottom of the table, together with a plot of the estimated polynomial:



Lag Distribution of M1	i	Coefficient	Std. Error	T-Statistic
	0	0.10270	0.14677	0.69970
	1	0.01159	0.10948	0.10587
	2	-0.00215	0.10139	-0.02123
	3	0.00920	0.06150	0.14955
	4	0.01766	0.07435	0.23756
	5	0.01363	0.06974	0.19547
	6	-4.7E-05	0.05557	-0.00084
	7	-0.01399	0.07080	-0.19764
	8	-0.01821	0.07537	-0.24158
	9	-0.00798	0.06399	-0.12475
	10	0.01017	0.10454	0.09726
	11	0.01260	0.11069	0.11386
	12	-0.04737	0.15693	-0.30182
Sum of Lags		0.08780	0.00297	29.5534

The Sum of Lags reported at the bottom of the table is the sum of the estimated coefficients on the distributed lag and has the interpretation of the long run effect of M1 on IP, assuming stationarity.

Note that selecting **View/Coefficient Tests** for an equation estimated with PDL terms tests the restrictions on γ, not on β. In this example, the coefficients on the fourth- (PDL05) and fifth-order (PDL06) terms are individually insignificant and very close to zero. To test

the joint significance of these two terms, click **View/Coefficient Tests/Wald-Coefficient Restrictions…** and enter:

```
c(6)=0, c(7)=0
```

in the Wald Test dialog box (see "Wald Test (Coefficient Restrictions)" on page 572 for an extensive discussion of Wald tests in EViews). EViews displays the result of the joint test:

Wald Test:
Equation: IP_PDL

Test Statistic	Value	df	Probability
F-statistic	0.039852	(2, 353)	0.9609
Chi-square	0.079704	2	0.9609

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
C(6)	2.58E-05	2448.827
C(7)	-4.93E-05	5550.537

Restrictions are linear in coefficients.

There is no evidence to reject the null hypothesis, suggesting that you could have fit a lower order polynomial to your lag structure.

Automatic Categorical Dummy Variables

EViews equation specifications support expressions of the form:

@EXPAND(*ser1[, ser2, ser3, …][, drop_spec]*)

that, when used in an equation specification, creates a set of dummy variables that span the unique integer or string values of the input series.

For example consider the following two variables:

- SEX is a numeric series which takes the values 1 and 0.
- REGION is an alpha series which takes the values "North", "South", "East", and "West".

The equation list specification

```
income age @expand(sex)
```

is used to regress INCOME on the regressor AGE, and two dummy variables, one for "SEX = 0" and one for "SEX = 1".

Similarly, the @EXPAND statement in the equation list specification,

```
income @expand(sex, region) age
```

creates 8 dummy variables corresponding to :

sex = 0, region = "North"

sex = 0, region = "South"

sex = 0, region = "East"

sex = 0, region = "West"

sex = 1, region = "North"

sex = 1, region = "South"

sex = 1, region = "East"

sex = 1, region = "West"

Note that our two example equation specifications did not include an intercept. This is because the default @EXPAND statements created a full set of dummy variables that would preclude including an intercept.

You may wish to drop one or more of the dummy variables. @EXPAND takes several options for dropping variables.

The option @DROPFIRST specifies that the first category should be dropped so that:

```
@expand(sex, region, @dropfirst)
```

no dummy is created for "SEX = 0, REGION = "North"".

Similarly, @DROPLAST specifies that the last category should be dropped. In:

```
@expand(sex, region, @droplast)
```

no dummy is created for "SEX = 1, REGION = "WEST"".

You may specify the dummy variables to be dropped, explicitly, using the modifier option @DROP(*val1[, val2, val3,...]*), where each argument specified corresponds to a successive category in @EXPAND. For example, in the expression:

```
@expand(sex, region, @drop(0,"West"), @drop(1,"North")
```

no dummy is created for "SEX = 0, REGION = "West"" and "SEX = 1, REGION = "North"".

When you specify drops by explicit value you may use the wild card "*" to indicate all values of a corresponding category. For example:

```
@expand(sex, region, @drop(1,*))
```

specifies that dummy variables for all values of REGION where "SEX = 1" should be dropped.

We caution you to take some care in using @EXPAND since it is very easy to generate excessively large numbers of regressors.

Example

Following Wooldridge (2000, Example 3.9, p. 106), we regress the log median housing price, LPRICE, on a constant, the log of the amount of pollution (LNOX), and the average number of houses in the community, ROOMS, using data from Harrison and Rubinfeld (1978).

We expand the example to include a dummy variable for each value of the series RADIAL, representing an index for community access to highways. We use @EXPAND to create the dummy variables of interest, with a list specification of:

```
lprice lnox rooms @expand(radial)
```

We deliberately omit the constant term C since the @EXPAND creates a full set of dummy variables. The top portion of the results is depicted below:

Dependent Variable: LPRICE
Method: Least Squares
Date: 12/30/03 Time: 16:49
Sample: 1 506
Included observations: 506

Variable	Coefficient	Std. Error	t-Statistic	Prob.
LNOX	-0.487579	0.084998	-5.736396	0.0000
ROOMS	0.284844	0.018790	15.15945	0.0000
RADIAL=1	8.930255	0.205986	43.35368	0.0000
RADIAL=2	9.030875	0.209225	43.16343	0.0000
RADIAL=3	9.085988	0.199781	45.47970	0.0000
RADIAL=4	8.960967	0.198646	45.11016	0.0000
RADIAL=5	9.110542	0.209759	43.43330	0.0000
RADIAL=6	9.001712	0.205166	43.87528	0.0000
RADIAL=7	9.013491	0.206797	43.58621	0.0000
RADIAL=8	9.070626	0.214776	42.23297	0.0000
RADIAL=24	8.811812	0.217787	40.46069	0.0000

Note that EViews has automatically created dummy variable expressions for each distinct value in RADIAL. If we wish to renormalize our dummy variables with respect to a different omitted category, we may include the C in the regression list, and explicitly exclude a value. For example, to exclude the category RADIAL = 24, we use the list:

```
lprice c lnox rooms @expand(radial, @drop(24))
```

Estimation of this specification yields:

Dependent Variable: LPRICE
Method: Least Squares
Date: 12/30/03 Time: 16:57
Sample: 1 506
Included observations: 506

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	8.811812	0.217787	40.46069	0.0000
LNOX	-0.487579	0.084998	-5.736396	0.0000
ROOMS	0.284844	0.018790	15.15945	0.0000
RADIAL=1	0.118444	0.072129	1.642117	0.1012
RADIAL=2	0.219063	0.066055	3.316398	0.0010
RADIAL=3	0.274176	0.059458	4.611253	0.0000
RADIAL=4	0.149156	0.042649	3.497285	0.0005
RADIAL=5	0.298730	0.037827	7.897337	0.0000
RADIAL=6	0.189901	0.062190	3.053568	0.0024
RADIAL=7	0.201679	0.077635	2.597794	0.0097
RADIAL=8	0.258814	0.066166	3.911591	0.0001

R-squared	0.573871	Mean dependent var	9.941057
Adjusted R-squared	0.565262	S.D. dependent var	0.409255
S.E. of regression	0.269841	Akaike info criterion	0.239530
Sum squared resid	36.04295	Schwarz criterion	0.331411
Log likelihood	-49.60111	F-statistic	66.66195
Durbin-Watson stat	0.671010	Prob(F-statistic)	0.000000

Weighted Least Squares

Suppose that you have heteroskedasticity of known form, and that there is a series w, whose values are proportional to the *reciprocals* of the error standard deviations. You can use weighted least squares, with weight series w, to correct for the heteroskedasticity.

EViews performs weighted least squares by first dividing the weight series by its mean, then multiplying all of the data for each observation by the scaled weight series. The scaling of the weight series is a normalization that has no effect on the parameter results, but makes the weighted residuals more comparable to the unweighted residuals. The normalization does imply, however, that EViews weighted least squares is not appropriate in situations where the scale of the weight series is relevant, as in frequency weighting.

Estimation is then completed by running a regression using the weighted dependent and independent variables to minimize the sum-of-squared residuals:

$$S(\beta) = \sum_t w_t^2 (y_t - x_t'\beta)^2 \tag{16.8}$$

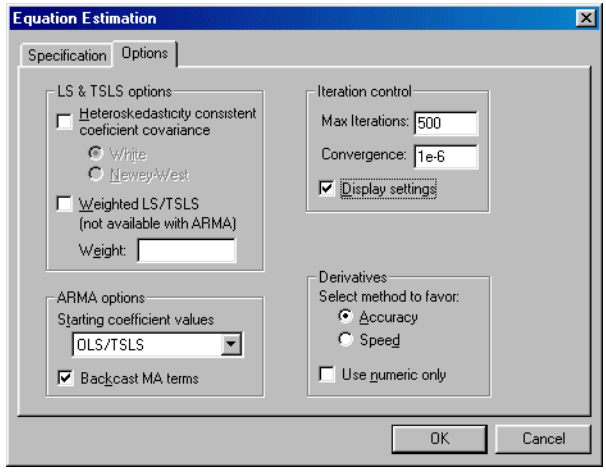
with respect to the k-dimensional vector of parameters β. In matrix notation, let W be a diagonal matrix containing the scaled w along the diagonal and zeroes elsewhere, and let y and X be the usual matrices associated with the left and right-hand side variables. The weighted least squares estimator is,

$$b_{WLS} = (X'W'WX)^{-1}X'W'Wy, \tag{16.9}$$

and the estimated covariance matrix is:

$$\hat{\Sigma}_{WLS} = s^2(X'W'WX)^{-1}. \tag{16.10}$$

To estimate an equation using weighted least squares, first go to the main menu and select **Quick/Estimate Equation...**, then choose **LS—Least Squares (NLS and ARMA)** from the combo box. Enter your equation specification and sample in the **Specification** tab, then select the **Options** tab and click on the **Weighted LS/TSLS** option.



Fill in the blank after **Weight** with the name of the series containing your weights, and click on **OK**. Click on **OK** again to accept the dialog and estimate the equation.

Dependent Variable: LOG(X)
Method: Least Squares
Date: 10/15/97 Time: 11:10
Sample(adjusted): 1891 1983
Included observations: 93 after adjusting endpoints
Weighting series: POP

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.004233	0.012745	0.332092	0.7406
LOG(X(-1))	0.099840	0.112539	0.887163	0.3774
LOG(W(-1))	0.194219	0.421005	0.461322	0.6457

Weighted Statistics

R-squared	0.016252	Mean dependent var	0.009762
Adjusted R-squared	-0.005609	S.D. dependent var	0.106487
S.E. of regression	0.106785	Akaike info criterion	-1.604274
Sum squared resid	1.026272	Schwarz criterion	-1.522577
Log likelihood	77.59873	F-statistic	0.743433
Durbin-Watson stat	1.948087	Prob(F-statistic)	0.478376

Unweighted Statistics

R-squared	-0.002922	Mean dependent var	0.011093
Adjusted R-squared	-0.025209	S.D. dependent var	0.121357
S.E. of regression	0.122877	Sum squared resid	1.358893
Durbin-Watson stat	2.086669		

EViews will open an output window displaying the standard coefficient results, and both weighted and unweighted summary statistics. The weighted summary statistics are based on the fitted residuals, computed using the weighted data:

$$\tilde{u}_t = w_t(y_t - x_t'b_{WLS}) . \tag{16.11}$$

The unweighted summary results are based on the residuals computed from the original (unweighted) data:

$$u_t = y_t - x_t'b_{WLS} . \tag{16.12}$$

Following estimation, the unweighted residuals are placed in the RESID series.

If the residual variance assumptions are correct, the weighted residuals should show no evidence of heteroskedasticity. If the variance assumptions are correct, the unweighted residuals should be heteroskedastic, with the reciprocal of the standard deviation of the residual at each period t being proportional to w_t.

The weighting option will be ignored in equations containing ARMA specifications. Note also that the weighting option is not available for binary, count, censored and truncated, or ordered discrete choice models.

Heteroskedasticity and Autocorrelation Consistent Covariances

When the form of heteroskedasticity is not known, it may not be possible to obtain efficient estimates of the parameters using weighted least squares. OLS provides consistent

parameter estimates in the presence of heteroskedasticity, but the usual OLS standard errors will be incorrect and should not be used for inference.

Before we describe the techniques for HAC covariance estimation, note that:

- Using the White heteroskedasticity consistent or the Newey-West HAC consistent covariance estimates does not change the point estimates of the parameters, only the estimated standard errors.
- There is nothing to keep you from combining various methods of accounting for heteroskedasticity and serial correlation. For example, weighted least squares estimation might be accompanied by White or Newey-West covariance matrix estimates.

Heteroskedasticity Consistent Covariances (White)

White (1980) has derived a heteroskedasticity consistent covariance matrix estimator which provides correct estimates of the coefficient covariances in the presence of heteroskedasticity of unknown form. The White covariance matrix is given by:

$$\hat{\Sigma}_W = \frac{T}{T-k}(X'X)^{-1}\left(\sum_{t=1}^{T} u_t^2 x_t x_t'\right)(X'X)^{-1}, \tag{16.13}$$

where is T the number of observations, k is the number of regressors, and u_t is the least squares residual.

EViews provides you the option to use the White covariance estimator in place of the standard OLS formula. Open the equation dialog and specify the equation as before, then push the **Options** button. Next, click on the check box labeled **Heteroskedasticity Consistent Covariance** and click on the **White** radio button. Accept the options and click **OK** to estimate the equation.

EViews will estimate your equation and compute the variances using White's covariance estimator. You can always tell when EViews is using White covariances, since the output display will include a line to document this fact:

Dependent Variable: LOG(X)
Method: Least Squares
Date: 10/15/97 Time: 11:11
Sample(adjusted): 1891 1983
Included observations: 93 after adjusting endpoints
Weighting series: POP
White Heteroskedasticity-Consistent Standard Errors & Covariance

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.004233	0.012519	0.338088	0.7361
LOG(X(-1))	0.099840	0.137262	0.727369	0.4689
LOG(W(-1))	0.194219	0.436644	0.444800	0.6575

HAC Consistent Covariances (Newey-West)

The White covariance matrix described above assumes that the residuals of the estimated equation are serially uncorrelated. Newey and West (1987) have proposed a more general covariance estimator that is consistent in the presence of both heteroskedasticity and autocorrelation of unknown form. The Newey-West estimator is given by,

$$\hat{\Sigma}_{NW} = \frac{T}{T-k}(X'X)^{-1}\hat{\Omega}(X'X)^{-1}, \tag{16.14}$$

where:

$$\hat{\Omega} = \frac{T}{T-k}\left\{\sum_{t=1}^{T} u_t^2 x_t x_t' + \sum_{v=1}^{q}\left(\left(1-\frac{v}{q+1}\right)\sum_{t=v+1}^{T}(x_t u_t u_{t-v} x_{t-v}' + x_{t-v}u_{t-v}u_t x_t')\right)\right\} \tag{16.15}$$

and q, the truncation lag, is a parameter representing the number of autocorrelations used in evaluating the dynamics of the OLS residuals u_t. Following the suggestion of Newey and West, EViews sets q using the formula:

$$q = \text{floor}(4(T/100)^{2/9}). \tag{16.16}$$

To use the Newey-West method, select the **Options** tab in the **Equation Estimation**. Check the box labeled **Heteroskedasticity Consistent Covariance** and press the **Newey-West** radio button.

Two-stage Least Squares

A fundamental assumption of regression analysis is that the right-hand side variables are uncorrelated with the disturbance term. If this assumption is violated, both OLS and weighted LS are biased and inconsistent.

There are a number of situations where some of the right-hand side variables are correlated with disturbances. Some classic examples occur when:

- There are endogenously determined variables on the right-hand side of the equation.
- Right-hand side variables are measured with error.

For simplicity, we will refer to variables that are correlated with the residuals as *endogenous*, and variables that are not correlated with the residuals as *exogenous* or *predetermined*.

The standard approach in cases where right-hand side variables are correlated with the residuals is to estimate the equation using *instrumental variables* regression. The idea

behind instrumental variables is to find a set of variables, termed *instruments*, that are both (1) correlated with the explanatory variables in the equation, and (2) uncorrelated with the disturbances. These instruments are used to eliminate the correlation between right-hand side variables and the disturbances.

Two-stage least squares (TSLS) is a special case of instrumental variables regression. As the name suggests, there are two distinct stages in two-stage least squares. In the first stage, TSLS finds the portions of the endogenous and exogenous variables that can be attributed to the instruments. This stage involves estimating an OLS regression of each variable in the model on the set of instruments. The second stage is a regression of the original equation, with all of the variables replaced by the fitted values from the first-stage regressions. The coefficients of this regression are the TSLS estimates.

You need not worry about the separate stages of TSLS since EViews will estimate both stages simultaneously using instrumental variables techniques. More formally, let Z be the matrix of instruments, and let y and X be the dependent and explanatory variables. Then the coefficients computed in two-stage least squares are given by,

$$b_{TSLS} = (X'Z(Z'Z)^{-1}Z'X)^{-1}X'Z(Z'Z)^{-1}Z'y, \tag{16.17}$$

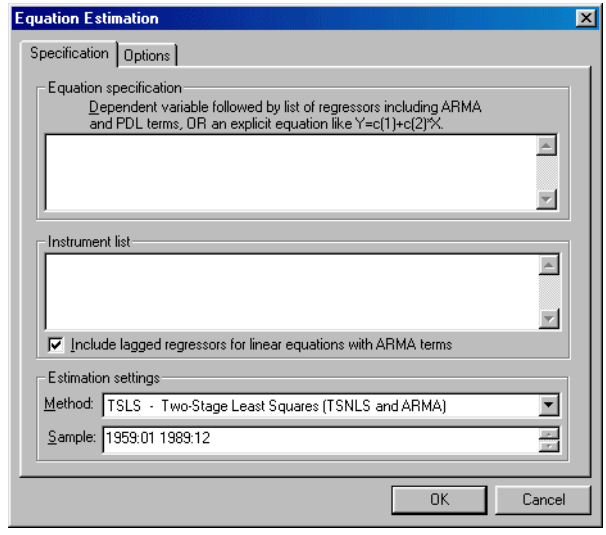
and the estimated covariance matrix of these coefficients is given by:

$$\hat{\Sigma}_{TSLS} = s^2(X'Z(Z'Z)^{-1}Z'X)^{-1}, \tag{16.18}$$

where s^2 is the estimated residual variance (square of the standard error of the regression).

Estimating TSLS in EViews

To use two-stage least squares, open the equation specification box by choosing **Object/New Object.../Equation...** or **Quick/Estimate Equation...**. Choose **TSLS** from the **Method:** combo box and the dialog will change to include an edit window where you will list the instruments.


Equation Estimation
Specification
Options
Equation specification
Dependent variable followed by list of regressors including ARMA and PDL terms, OR an explicit equation like Y=c(1)+c(2)*X.
Instrument list
Include lagged regressors for linear equations with ARMA terms
Estimation settings
Method: TSLS - Two-Stage Least Squares (TSNLS and ARMA)
Sample: 1959:01 1989:12
OK
Cancel

In the **Equation specification** edit box, specify your dependent variable and independent variables, and in the **Instrument list** edit box, provide a list of instruments.

There are a few things to keep in mind as you enter your instruments:

- In order to calculate TSLS estimates, your specification must satisfy the *order condition* for identification, which says that there must be at least as many instruments as there are coefficients in your equation. There is an additional rank condition which must also be satisfied. See Davidson and MacKinnon (1994) and Johnston and DiNardo (1997) for additional discussion.
- For econometric reasons that we will not pursue here, any right-hand side variables that are not correlated with the disturbances should be included as instruments.
- The constant, C, is always a suitable instrument, so EViews will add it to the instrument list if you omit it.

For example, suppose you are interested in estimating a consumption equation relating consumption (CONS) to gross domestic product (GDP), lagged consumption (CONS(–1)), a trend variable (TIME) and a constant (C). GDP is endogenous and therefore correlated with the residuals. You may, however, believe that government expenditures (G), the log of the money supply (LM), lagged consumption, TIME, and C, are exogenous and uncorrelated with the disturbances, so that these variables may be used as instruments. Your equation specification is then,

```
cons c gdp cons(-1) time
```

and the instrument list is:

```
c gov cons(-1) time lm
```

This specification satisfies the order condition for identification, which requires that there are at least as many instruments (five) as there are coefficients (four) in the equation specification.

Furthermore, all of the variables in the consumption equation that are believed to be uncorrelated with the disturbances, (CONS(–1), TIME, and C), appear both in the equation specification and in the instrument list. Note that listing C as an instrument is redundant, since EViews automatically adds it to the instrument list.

Output from TSLS

Below, we present TSLS estimates from a regression of LOG(CS) on a constant and LOG(GDP), with the instrument list "C LOG(CS(-1)) LOG(GDP(-1))":

Dependent Variable: LOG(CS)
Method: Two-Stage Least Squares
Date: 10/15/97 Time: 11:32
Sample(adjusted): 1947:2 1995:1
Included observations: 192 after adjusting endpoints
Instrument list: C LOG(CS(-1)) LOG(GDP(-1))

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-1.209268	0.039151	-30.88699	0.0000
LOG(GDP)	1.094339	0.004924	222.2597	0.0000

R-squared	0.996168	Mean dependent var	7.480286
Adjusted R-squared	0.996148	S.D. dependent var	0.462990
S.E. of regression	0.028735	Sum squared resid	0.156888
F-statistic	49399.36	Durbin-Watson stat	0.102639
Prob(F-statistic)	0.000000		

EViews identifies the estimation procedure, as well as the list of instruments in the header. This information is followed by the usual coefficient, t-statistics, and asymptotic p-values.

The summary statistics reported at the bottom of the table are computed using the formulas outlined in "Summary Statistics" on page 451. Bear in mind that all reported statistics are only asymptotically valid. For a discussion of the finite sample properties of TSLS, see Johnston and DiNardo (1997, pp. 355–358) or Davidson and MacKinnon (1984, pp. 221–224).

EViews uses the *structural residuals* $u_t = y_t - x_t'b_{TSLS}$ in calculating all of the summary statistics. For example, the standard error of the regression used in the asymptotic covariance calculation is computed as:

$$s^2 = \sum_t u_t^2/(T-k)\,. \tag{16.19}$$

These structural residuals should be distinguished from the *second stage residuals* that you would obtain from the second stage regression if you actually computed the two-stage least squares estimates in two separate stages. The second stage residuals are given by $\tilde{u}_t = \hat{y}_t - \hat{x}_t'b_{TSLS}$, where the $\hat{y}_t$ and $\hat{x}_t$ are the fitted values from the first-stage regressions.

We caution you that some of the reported statistics should be interpreted with care. For example, since different equation specifications will have different instrument lists, the reported R^2 for TSLS can be negative even when there is a constant in the equation.

Weighted TSLS

You can combine TSLS with weighted regression. Simply enter your TSLS specification as above, then press the **Options** button, select the **Weighted LS/TSLS** option, and enter the weighting series.

Weighted two-stage least squares is performed by multiplying all of the data, including the instruments, by the weight variable, and estimating TSLS on the transformed model. Equivalently, EViews then estimates the coefficients using the formula,

$$b_{WTSLS} = (X'W'WZ(Z'W'WZ)^{-1}Z'W'WX)^{-1} \cdot X'W'WZ(Z'W'WZ)^{-1}Z'W'Wy \tag{16.20}$$

The estimated covariance matrix is:

$$\hat{\Sigma}_{WTSLS} = s^2(X'W'WZ(Z'W'WZ)^{-1}Z'W'WX)^{-1}. \tag{16.21}$$

TSLS with AR errors

You can adjust your TSLS estimates to account for serial correlation by adding AR terms to your equation specification. EViews will automatically transform the model to a nonlinear least squares problem, and estimate the model using instrumental variables. Details of this procedure may be found in Fair (1984, pp. 210–214). The output from TSLS with an AR(1) specification looks as follows:

Dependent Variable: LOG(CS)
Method: Two-Stage Least Squares
Date: 10/15/97 Time: 11:42
Sample(adjusted): 1947:2 1995:1
Included observations: 192 after adjusting endpoints
Convergence achieved after 4 iterations
Instrument list: C LOG(CS(-1)) LOG(GDP(-1))

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-1.420705	0.203266	-6.989390	0.0000
LOG(GDP)	1.119858	0.025116	44.58782	0.0000
AR(1)	0.930900	0.022267	41.80595	0.0000

R-squared	0.999611	Mean dependent var	7.480286
Adjusted R-squared	0.999607	S.D. dependent var	0.462990
S.E. of regression	0.009175	Sum squared resid	0.015909
F-statistic	243139.7	Durbin-Watson stat	1.931027
Prob(F-statistic)	0.000000		
Inverted AR Roots	.93		

The **Options** button in the estimation box may be used to change the iteration limit and convergence criterion for the nonlinear instrumental variables procedure.

First-order AR errors

Suppose your specification is:

$$\begin{aligned} y_t &= x_t'\beta + w_t\gamma + u_t \\ u_t &= \rho_1 u_{t-1} + \epsilon_t \end{aligned} \tag{16.22}$$

where x_t is a vector of endogenous variables, and w_t is a vector of predetermined variables, which, in this context, may include lags of the dependent variable. z_t is a vector of instrumental variables not in w_t that is large enough to identify the parameters of the model.

In this setting, there are important technical issues to be raised in connection with the choice of instruments. In a widely quoted result, Fair (1970) shows that if the model is estimated using an iterative Cochrane-Orcutt procedure, all of the lagged left- and right-hand side variables $(y_{t-1}, x_{t-1}, w_{t-1})$ must be included in the instrument list to obtain consistent estimates. In this case, then the instrument list should include:

$$(w_t, z_t, y_{t-1}, x_{t-1}, w_{t-1}) . \tag{16.23}$$

Despite the fact the EViews estimates the model as a nonlinear regression model, the first stage instruments in TSLS are formed as if running Cochrane-Orcutt. Thus, if you choose to omit the lagged left- and right-hand side terms from the instrument list, EViews will automatically add each of the lagged terms as instruments. This fact is noted in your output.

Higher Order AR errors

The AR(1) result extends naturally to specifications involving higher order serial correlation. For example, if you include a single AR(4) term in your model, the natural instrument list will be:

$$(w_t, z_t, y_{t-4}, x_{t-4}, w_{t-4}) \tag{16.24}$$

If you include AR terms from 1 through 4, one possible instrument list is:

$$(w_t, z_t, y_{t-1}, \ldots, y_{t-4}, x_{t-1}, \ldots, x_{t-4}, w_{t-1}, \ldots, w_{t-4}) \tag{16.25}$$

Note that while theoretically valid, this instrument list has a large number of overidentifying instruments, which may lead to computational difficulties and large finite sample biases (Fair (1984, p. 214), Davidson and MacKinnon (1993, pp. 222-224)). In theory, adding instruments should always improve your estimates, but as a practical matter this may not be so in small samples.

Examples

Suppose that you wish to estimate the consumption function by two-stage least squares, allowing for first-order serial correlation. You may then use two-stage least squares with the variable list,

```
cons c gdp ar(1)
```

and instrument list:

```
c gov log(m1) time cons(-1) gdp(-1)
```

Notice that the lags of both the dependent and endogenous variables (CONS(–1) and GDP(–1)), are included in the instrument list.

Similarly, consider the consumption function:

```
cons c cons(-1) gdp ar(1)
```

A valid instrument list is given by:

```
c gov log(m1) time cons(-1) cons(-2) gdp(-1)
```

Here we treat the lagged left and right-hand side variables from the original specification as predetermined and add the lagged values to the instrument list.

Lastly, consider the specification:

```
cons c gdp ar(1) ar(2) ar(3) ar(4)
```

Adding all of the relevant instruments in the list, we have:

```
c gov log(m1) time cons(-1) cons(-2) cons(-3) cons(-4) gdp(-1)
   gdp(-2) gdp(-3) gdp(-4)
```

TSLS with MA errors

You can also estimate two-stage least squares variable problems with MA error terms of various orders. To account for the presence of MA errors, simply add the appropriate terms to your specification prior to estimation.

Illustration

Suppose that you wish to estimate the consumption function by two-stage least squares, accounting for first-order moving average errors. You may then use two-stage least squares with the variable list,

```
cons c gdp ma(1)
```

and instrument list:

```
c gov log(m1) time
```

EViews will add both first and second lags of CONS and GDP to the instrument list.

Technical Details

Most of the technical details are identical to those outlined above for AR errors. EViews transforms the model that is nonlinear in parameters (employing backcasting, if appropriate) and then estimates the model using nonlinear instrumental variables techniques.

Note that EViews augments the instrument list by adding lagged dependent and regressor variables. Note however, that each MA term involves an infinite number of AR terms. Clearly, it is impossible to add an infinite number of lags to the instrument list, so that EViews performs an ad hoc approximation by adding a truncated set of instruments involving the MA order and an additional lag. If for example, you have an MA(5), EViews will add lagged instruments corresponding to lags 5 and 6.

Nonlinear Least Squares

Suppose that we have the regression specification:

$$y_t = f(x_t, \beta) + \epsilon_t, \tag{16.26}$$

where f is a general function of the explanatory variables x_t and the parameters β. Least squares estimation chooses the parameter values that minimize the sum of squared residuals:

$$S(\beta) = \sum_t (y_t - f(x_t, \beta))^2 = (y - f(X, \beta))'(y - f(X, \beta)) \tag{16.27}$$

We say that a model is *linear in parameters* if the derivatives of f with respect to the parameters do not depend upon β; if the derivatives are functions of β, we say that the model is *nonlinear in parameters*.

For example, consider the model given by:

$$y_t = \beta_1 + \beta_2 \log L_t + \beta_3 \log K_t + \epsilon_t. \tag{16.28}$$

It is easy to see that this model is linear in its parameters, implying that it can be estimated using ordinary least squares.

In contrast, the equation specification:

$$y_t = \beta_1 L_t^{\beta_2} K_t^{\beta_3} + \epsilon_t \tag{16.29}$$

has derivatives that depend upon the elements of β. There is no way to rearrange the terms in this model so that ordinary least squares can be used to minimize the sum-of-squared residuals. We must use nonlinear least squares techniques to estimate the parameters of the model.

Nonlinear least squares minimizes the sum-of-squared residuals with respect to the choice of parameters β. While there is no closed form solution for the parameter estimates, the estimates satisfy the first-order conditions:

$$(G(\beta))'(y - f(X, \beta)) = 0, \tag{16.30}$$

where $G(\beta)$ is the matrix of first derivatives of $f(X, \beta)$ with respect to β (to simplify notation we suppress the dependence of G upon X). The estimated covariance matrix is given by:

$$\hat{\Sigma}_{NLLS} = s^2(G(b_{NLLS})'G(b_{NLLS}))^{-1}. \tag{16.31}$$

where b_{NLLS} are the estimated parameters. For additional discussion of nonlinear estimation, see Pindyck and Rubinfeld (1991, pp. 231-245) or Davidson and MacKinnon (1993).

Estimating NLS Models in EViews

It is easy to tell EViews that you wish to estimate the parameters of a model using nonlinear least squares. EViews automatically applies nonlinear least squares to any regression equation that is nonlinear in its coefficients. Simply select **Object/New Object.../Equation**, enter the equation in the equation specification dialog box, and click **OK**. EViews will do all of the work of estimating your model using an iterative algorithm.

A full technical discussion of iterative estimation procedures is provided in Appendix C, "Estimation and Solution Options", beginning on page 951.

Specifying Nonlinear Least Squares

For nonlinear regression models, you will have to enter your specification in equation form using EViews expressions that contain direct references to coefficients. You may use elements of the default coefficient vector C (e.g. C(1), C(2), C(34), C(87)), or you can define and use other coefficient vectors. For example:

```
y = c(1) + c(2)*(k^c(3)+l^c(4))
```

is a nonlinear specification that uses the first through the fourth elements of the default coefficient vector, C.

To create a new coefficient vector, select **Object/New Object.../Matrix-Vector-Coef/Coefficient Vector** in the main menu and provide a name. You may now use this coefficient vector in your specification. For example, if you create a coefficient vector named CF, you can rewrite the specification above as:

```
y = cf(11) + cf(12)*(k^cf(13)+l^cf(14))
```

which uses the eleventh through the fourteenth elements of CF.

You can also use multiple coefficient vectors in your specification:

```
y = c(11) + c(12)*(k^cf(1)+l^cf(2))
```

which uses both C and CF in the specification.

It is worth noting that EViews implicitly adds an additive disturbance to your specification. For example, the input

```
y = (c(1)*x + c(2)*z + 4)^2
```

is interpreted as $y_t = (c(1)x_t + c(2)z_t + 4)^2 + \epsilon_t$, and EViews will minimize:

$$S(c(1), c(2)) = \sum_t (y_t - (c(1)x_t + c(2)z_t + 4)^2)^2 \tag{16.32}$$

If you wish, the equation specification may be given by a simple expression that does not include a dependent variable. For example, the input,

```
(c(1)*x + c(2)*z + 4)^2
```

is interpreted by EViews as $-(c(1)x_t + c(2)z_t + 4)^2 = \epsilon_t$, and EViews will minimize:

$$S(c(1), c(2)) = \sum_t (-(c(1)x_t + c(2)z_t + 4)^2)^2 \tag{16.33}$$

While EViews will estimate the parameters of this last specification, the equation cannot be used for forecasting and cannot be included in a model. This restriction also holds for any equation that includes coefficients to the left of the equal sign. For example, if you specify,

```
x + c(1)*y = z^c(2)
```

EViews will find the values of C(1) and C(2) that minimize the sum of squares of the implicit equation:

$$x_t + c(1)y_t - z_t^{c(2)} = \epsilon_t \tag{16.34}$$

The estimated equation cannot be used in forecasting or included in a model, since there is no dependent variable.

Estimation Options

Starting Values. Iterative estimation procedures require starting values for the coefficients of the model. There are no general rules for selecting starting values for parameters. The closer to the true values the better, so if you have reasonable guesses for parameter values, these can be useful. In some cases, you can obtain good starting values by estimating a restricted version of the model using least squares. In general, however, you will have to experiment in order to find starting values.

EViews uses the values in the coefficient vector at the time you begin the estimation procedure as starting values for the iterative procedure. It is easy to examine and change these coefficient starting values.

To see the starting values, double click on the coefficient vector in the workfile directory. If the values appear to be reasonable, you can close the window and proceed with estimating your model.

If you wish to change the starting values, first make certain that the spreadsheet view of your coefficients is in edit mode, then enter the coefficient values. When you are finished setting the initial values, close the coefficient vector window and estimate your model.

You may also set starting coefficient values from the command window using the PARAM command. Simply enter the PARAM keyword, following by each coefficient and desired value:

```
param c(1) 153 c(2) .68 c(3) .15
```

sets C(1) = 153, C(2) = .68, and C(3) = .15.

See Appendix C, "Estimation and Solution Options" on page 951, for further details.

Derivative Methods. Estimation in EViews requires computation of the derivatives of the regression function with respect to the parameters. EViews provides you with the option of computing analytic expressions for these derivatives (if possible), or computing finite difference numeric derivatives in cases where the derivative is not constant. Furthermore, if numeric derivatives are computed, you can choose whether to favor speed of computation (fewer function evaluations) or whether to favor accuracy (more function evaluations). Additional issues associated with ARIMA models are discussed in "Estimation Options" on page 509.

Iteration and Convergence Options. You can control the iterative process by specifying convergence criterion and the maximum number of iterations. Press the **Options** button in the equation dialog box and enter the desired values.

EViews will report that the estimation procedure has converged if the convergence test value is below your convergence tolerance. See "Iteration and Convergence Options" on page 953 for details.

In most cases, you will not need to change the maximum number of iterations. However, for some difficult to estimate models, the iterative procedure will not converge within the maximum number of iterations. If your model does not converge within the allotted number of iterations, simply click on the **Estimate** button, and, if desired, increase the maximum number of iterations. Click on **OK** to accept the options, and click on **OK** to begin estimation. EViews will start estimation using the last set of parameter values as starting values.

These options may also be set from the global options dialog. See Appendix A, "Estimation Defaults" on page 941.

Output from NLS

Once your model has been estimated, EViews displays an equation output screen showing the results of the nonlinear least squares procedure. Below is the output from a regression of LOG(CS) on C, and the Box-Cox transform of GDP:

Dependent Variable: LOG(CS)
Method: Least Squares
Date: 10/15/97 Time: 11:51
Sample(adjusted): 1947:1 1995:1
Included observations: 193 after adjusting endpoints
Convergence achieved after 80 iterations
LOG(CS)= C(1)+C(2)*(GDP^C(3)-1)/C(3)

	Coefficient	Std. Error	t-Statistic	Prob.
C(1)	2.851780	0.279033	10.22024	0.0000
C(2)	0.257592	0.041147	6.260254	0.0000
C(3)	0.182959	0.020201	9.056824	0.0000

R-squared	0.997252	Mean dependent var	7.476058
Adjusted R-squared	0.997223	S.D. dependent var	0.465503
S.E. of regression	0.024532	Akaike info criterion	-4.562220
Sum squared resid	0.114350	Schwarz criterion	-4.511505
Log likelihood	443.2542	F-statistic	34469.84
Durbin-Watson stat	0.134628	Prob(F-statistic)	0.000000

If the estimation procedure has converged, EViews will report this fact, along with the number of iterations that were required. If the iterative procedure did not converge, EViews will report "Convergence not achieved after" followed by the number of iterations attempted.

Below the line describing convergence, EViews will repeat the nonlinear specification so that you can easily interpret the estimated coefficients of your model.

EViews provides you with all of the usual summary statistics for regression models. Provided that your model has converged, the standard statistical results and tests are *asymptotically* valid.

Weighted NLS

Weights can be used in nonlinear estimation in a manner analogous to weighted linear least squares. To estimate an equation using weighted nonlinear least squares, enter your specification, press the **Options** button and click on the **Weighted LS/TSLS** option. Fill in the blank after **Weight:** with the name of the weight series and then estimate the equation.

EViews minimizes the sum of the weighted squared residuals:

$$S(\beta) = \sum_t w_t^2 (y_t - f(x_t, \beta))^2 = (y - f(X, \beta))' W' W (y - f(X, \beta)) \quad (16.35)$$

with respect to the parameters β, where w_t are the values of the weight series and W is the matrix of weights. The first-order conditions are given by,

$$(G(\beta))'W'W(y - f(X, \beta)) = 0 \tag{16.36}$$

and the covariance estimate is computed as:

$$\hat{\Sigma}_{WNLLS} = s^2(G(b_{WNLLS})'W'WG(b_{WNLLS}))^{-1}. \tag{16.37}$$

NLS with AR errors

EViews will estimate nonlinear regression models with autoregressive error terms. Simply select **Object/New Object.../Equation...** or **Quick/Estimate Equation...** and specify your model using EViews expressions, followed by an additive term describing the AR correction enclosed in square brackets. The AR term should consist of a coefficient assignment for each AR term, separated by commas. For example, if you wish to estimate,

$$\begin{aligned} CS_t &= c_1 + GDP_t^{c_2} + u_t \\ u_t &= c_3 u_{t-1} + c_4 u_{t-2} + \epsilon_t \end{aligned} \tag{16.38}$$

you should enter the specification:

```
cs = c(1) + gdp^c(2) + [ar(1)=c(3), ar(2)=c(4)]
```

See "How EViews Estimates AR Models" on page 500 for additional details. EViews does not currently estimate nonlinear models with MA errors, nor does it estimate weighted models with AR terms—if you add AR terms to a weighted nonlinear model, the weighting series will be ignored.

Nonlinear TSLS

Nonlinear two-stage least squares refers to an instrumental variables procedure for estimating nonlinear regression models involving functions of endogenous and exogenous variables and parameters. Suppose we have the usual nonlinear regression model:

$$y_t = f(x_t, \beta) + \epsilon_t, \tag{16.39}$$

where β is a k-dimensional vector of parameters, and x_t contains both exogenous and endogenous variables. In matrix form, if we have $m \geq k$ instruments z_t, nonlinear two-stage least squares minimizes:

$$S(\beta) = (y - f(X, \beta))'Z(Z'Z)^{-1}Z'(y - f(X, \beta)) \tag{16.40}$$

with respect to the choice of β.

While there is no closed form solution for the parameter estimates, the parameter estimates satisfy the first-order conditions:

$$G(\beta)'Z(Z'Z)^{-1}Z'(y - f(X, \beta)) = 0 \tag{16.41}$$

with estimated covariance given by:

$$\hat{\Sigma}_{TSNLLS} = s^2(G(b_{TSNLLS})'Z(Z'Z)^{-1}Z'G(b_{TSNLLS}))^{-1}. \quad (16.42)$$

How to Estimate Nonlinear TSLS in EViews

EViews performs the estimation procedure in a single step so that you don't have to perform the separate stages yourself. Simply select **Object/New Object.../Equation...** or **Quick/Estimate Equation...** Choose **TSLS** from the **Method:** combo box, enter your nonlinear specification and the list of instruments. Click **OK**.

With nonlinear two-stage least squares estimation, you have a great deal of flexibility with your choice of instruments. Intuitively, you want instruments that are correlated with $G(\beta)$. Since G is nonlinear, you may begin to think about using more than just the exogenous and predetermined variables as instruments. Various nonlinear functions of these variables, for example, cross-products and powers, may also be valid instruments. One should be aware, however, of the possible finite sample biases resulting from using too many instruments.

Weighted Nonlinear Two-stage Least Squares

Weights can be used in nonlinear two-stage least squares estimation. Simply add weighting to your nonlinear TSLS specification above by pressing the **Options** button, selecting **Weighted LS/TSLS** option, and entering the name of the weight series.

The objective function for weighted TSLS is,

$$S(\beta) = (y - f(X, \beta))'W'WZ(Z'W'WZ)^{-1}Z'W'W(y - f(X, \beta)). \quad (16.43)$$

The reported standard errors are based on the covariance matrix estimate given by:

$$\hat{\Sigma}_{WTSNLLS} = s^2(G(b)'W'WZ(Z'W'WZ)^{-1}Z'W'WG(b))^{-1} \quad (16.44)$$

where $b \equiv b_{WTSNLLS}$. Note that if you add AR or MA terms to a weighted specification, the weighting series will be ignored.

Nonlinear Two-stage Least Squares with AR errors

While we will not go into much detail here, note that EViews can estimate non-linear TSLS models where there are autoregressive error terms. EViews does not currently estimate nonlinear models with MA errors.

To estimate your model, simply open your equation specification window, and enter your nonlinear specification, including all AR terms, and provide your instrument list. For example, you could enter the regression specification:

```
cs = exp(c(1) + gdp^c(2)) + [ar(1)=c(3)]
```

with the instrument list:

```
c gov
```

EViews will transform the nonlinear regression model as described in "Estimating AR Models" on page 497, and then estimate nonlinear TSLS on the transformed specification using the instruments C and GOV. For nonlinear models with AR errors, EViews uses a Gauss-Newton algorithm. See "Optimization Algorithms" on page 956 for further details.

Solving Estimation Problems

EViews may not be able to estimate your nonlinear equation on the first attempt. Sometimes, the nonlinear least squares procedure will stop immediately. Other times, EViews may stop estimation after several iterations without achieving convergence. EViews might even report that it cannot improve the sums-of-squares. While there are no specific rules on how to proceed if you encounter these estimation problems, there are a few general areas you might want to examine.

Starting Values

If you experience problems with the very first iteration of a nonlinear procedure, the problem is almost certainly related to starting values. See the discussion above for how to examine and change your starting values.

Model Identification

If EViews goes through a number of iterations and then reports that it encounters a "Near Singular Matrix", you should check to make certain that your model is identified. Models are said to be non-identified if there are multiple sets of coefficients which identically yield the minimized sum-of-squares value. If this condition holds, it is impossible to choose between the coefficients on the basis of the minimum sum-of-squares criterion.

For example, the nonlinear specification:

$$y_t = \beta_1\beta_2 + \beta_2^2 x_t + \epsilon_t \tag{16.45}$$

is not identified, since any coefficient pair (β_1, β_2) is indistinguishable from the pair $(-\beta_1, -\beta_2)$ in terms of the sum-of-squared residuals.

For a thorough discussion of identification of nonlinear least squares models, see Davidson and MacKinnon (1993, Sections 2.3, 5.2 and 6.3).

Convergence Criterion

EViews may report that it is unable to improve the sums-of-squares. This result may be evidence of non-identification or model misspecification. Alternatively, it may be the result of

setting your convergence criterion too low, which can occur if your nonlinear specification is particularly complex.

If you wish to change the convergence criterion, enter the new value in the **Options** tab. Be aware that increasing this value increases the possibility that you will stop at a local minimum, and may hide misspecification or non-identification of your model.

See "Setting Estimation Options" on page 951, for related discussion.

Generalized Method of Moments (GMM)

The starting point of GMM estimation is a theoretical relation that the parameters should satisfy. The idea is to choose the parameter estimates so that the theoretical relation is satisfied as "closely" as possible. The theoretical relation is replaced by its sample counterpart and the estimates are chosen to minimize the weighted distance between the theoretical and actual values. GMM is a robust estimator in that, unlike maximum likelihood estimation, it does not require information of the exact distribution of the disturbances. In fact, many common estimators in econometrics can be considered as special cases of GMM.

The theoretical relation that the parameters should satisfy are usually *orthogonality conditions* between some (possibly nonlinear) function of the parameters $f(\theta)$ and a set of instrumental variables z_t:

$$\mathrm{E}(f(\theta)'Z) = 0\,, \tag{16.46}$$

where θ are the parameters to be estimated. The GMM estimator selects parameter estimates so that the sample correlations between the instruments and the function f are as close to zero as possible, as defined by the criterion function:

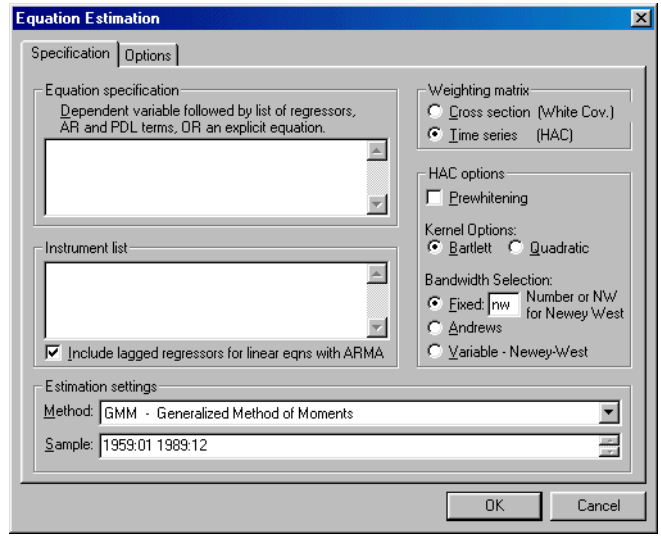
$$J(\theta) = (m(\theta))'Am(\theta)\,, \tag{16.47}$$

where $m(\theta) = f(\theta)'Z$ and A is a weighting matrix. Any symmetric positive definite matrix A will yield a consistent estimate of q. However, it can be shown that a necessary (but not sufficient) condition to obtain an (asymptotically) efficient estimate of q is to set A equal to the inverse of the covariance matrix of the sample moments m. It is worth noting here that in standard single equation GMM, EViews estimates a model with the objective function (16.47) divided by the number of observations. For equations with panel data ("GMM Estimation" beginning on page 905) the objective function is *not* similarly scaled.

Many standard estimators, including all of the system estimators provided in EViews, can be set up as special cases of GMM. For example, the ordinary least squares estimator can be viewed as a GMM estimator, based upon the conditions that each of the right-hand variables is uncorrelated with the residual.

Estimation by GMM in EViews

To estimate an equation by GMM, either create a new equation object by selecting **Object/New Object.../Equation**, or press the **Estimate** button in the toolbar of an existing equation. From the **Equation Specification** dialog choose **Estimation Method: GMM**. The estimation specification dialog will change as depicted below.



To obtain GMM estimates in EViews, you need to write the moment condition as an orthogonality condition between an expression including the parameters and a set of instrumental variables. There are two ways you can write the orthogonality condition: with and without a dependent variable.

If you specify the equation either by listing variable names or by an expression with an equal sign, EViews will interpret the moment condition as an orthogonality condition between the instruments and the residuals defined by the equation. If you specify the equation by an expression without an equal sign, EViews will orthogonalize that expression to the set of instruments.

You must also list the names of the instruments in the **Instrument list** field box of the Equation Specification dialog box. For the GMM estimator to be identified, there must be at least as many instrumental variables as there are parameters to estimate. EViews will always include the constant in the list of instruments.

For example, if you type,

Equation Specification: `y c x`

Instrument list: `c z w`

the orthogonality conditions given by:

$$\begin{aligned} \sum(y_t - c(1) - c(2)x_t) &= 0 \\ \sum(y_t - c(1) - c(2)x_t)z_t &= 0 \\ \sum(y_t - c(1) - c(2)x_t)w_t &= 0 \end{aligned} \tag{16.48}$$

If you enter the specification,

Equation Specification: `c(1)*log(y)+x^c(2)`

Instrument list: `c z z(-1)`

the orthogonality conditions are:

$$\begin{aligned} \sum(c(1)\log y_t + x_t^{c(2)}) &= 0 \\ \sum(c(1)\log y_t + x_t^{c(2)})z_t &= 0 \\ \sum(c(1)\log y_t + x_t^{c(2)})z_{t-1} &= 0 \end{aligned} \tag{16.49}$$

On the right part of the Equation Specification dialog are the options for selecting the weighting matrix A in the objective function. If you select **Weighting Matrix: Cross section (White Cov)**, the GMM estimates will be robust to heteroskedasticity of unknown form.

If you select **Weighting Matrix: Time series (HAC)**, the GMM estimates will be robust to heteroskedasticity and autocorrelation of unknown form. For the HAC option, you have to specify the kernel type and bandwidth.

- The **Kernel Options** determine the functional form of the kernel used to weight the autocovariances in computing the weighting matrix.
- The **Bandwidth Selection** option determines how the weights given by the kernel change with the lags of the autocovariances in the computation of the weighting matrix. If you select **Fixed** bandwidth, you may either enter a number for the bandwidth or type `nw` to use Newey and West's fixed bandwidth selection criterion.
- The **Prewhitening** option runs a preliminary VAR(1) prior to estimation to "soak up" the correlation in the moment conditions.

The technical notes in "Generalized Method of Moments (GMM)" on page 716 describe these options in greater detail.

Example

Tauchen (1986) considers the problem of estimating the taste parameters β, γ from the Euler equation:

$$(\beta R_{t+1} w_{t+1}^{-\gamma} - 1)' z_t = 0 \tag{16.50}$$

where we use instruments $z_t = (1, w_t, w_{t-1}, r_t, r_{t-1})'$. To estimate the parameters β, γ by GMM, fill in the Equation Specification dialog as:

Equation Specification: `c(1)*r(+1)*w(+1)^(-c(2))-1`

Instrument list: `c w w(-1) r r(-1)`

The estimation result using the default HAC **Weighting Matrix** option looks as follows:

Dependent Variable: Implicit Equation
Method: Generalized Method of Moments
Date: 09/26/97 Time: 14:02
Sample(adjusted): 1891 1982
Included observations: 92 after adjusting endpoints
No prewhitening
Bandwidth: Fixed (3)
Kernel: Bartlett
Convergence achieved after: 7 weight matricies, 7 total coef iterations
C(1)*R(+1)*W(+1)^(-C(2))-1
Instrument list: C W W(-1) R R(-1)

	Coefficient	Std. Error	t-Statistic	Prob.
C(1)	0.934096	0.018751	49.81600	0.0000
C(2)	1.366396	0.741802	1.841995	0.0688
S.E. of regression	0.154084	Sum squared resid		2.136760
Durbin-Watson stat	1.903837	J-statistic		0.054523

Note that when you specify an equation without a dependent variable, EViews does not report some of the regression statistics such as the R-squared. The J-statistic reported at the bottom of the table is the minimized value of the objective function, where we report (16.47) divided by the number of observations (see "Generalized Method of Moments (GMM)" beginning on page 488 for additional discussion). This J-statistic may be used to carry out hypothesis tests from GMM estimation; see Newey and West (1987a). A simple application of the J-statistic is to test the validity of overidentifying restrictions when you have more instruments than parameters to estimate. In this example, we have five instruments to estimate two parameters and so there are three overidentifying restrictions. Under the null hypothesis that the overidentifying restrictions are satisfied, the J-statistic times the number of regression observations is asymptotically χ^2 with degrees of freedom equal to the number of overidentifying restrictions. You can compute the test statistic as a named scalar in EViews using the commands:

```
scalar overid=eq_gmm.@regobs*eq_gmm.@jstat
scalar overid_p=1-@cchisq(overid,3)
```

where EQ_GMM is the name of the equation containing the GMM estimates. The second command computes the p-value of the test statistic as a named scalar OVERID_P. To view the value of OVERID_P, double click on its name; the value will be displayed in the status line at the bottom of the EViews window.

Chapter 17. Time Series Regression

In this section, we discuss single equation regression techniques that are important for the analysis of time series data: testing for serial correlation, estimation of ARMA models, using polynomial distributed lags, and testing for unit roots in potentially nonstationary time series.

The chapter focuses on the specification and estimation of time series models. A number of related topics are discussed elsewhere. For example, standard multiple regression techniques are discussed in Chapter 15, "Basic Regression", on page 443 and Chapter 16, "Additional Regression Methods", on page 461, while forecasting and inference are discussed extensively in Chapter 18, "Forecasting from an Equation", on page 543.

Serial Correlation Theory

A common finding in time series regressions is that the residuals are correlated with their own lagged values. This serial correlation violates the standard assumption of regression theory that disturbances are not correlated with other disturbances. The primary problems associated with serial correlation are:

- OLS is no longer efficient among linear estimators. Furthermore, since prior residuals help to predict current residuals, we can take advantage of this information to form a better prediction of the dependent variable.
- Standard errors computed using the textbook OLS formula are not correct, and are generally understated.
- If there are lagged dependent variables on the right-hand side, OLS estimates are biased and inconsistent.

EViews provides tools for detecting serial correlation and estimation methods that take account of its presence.

In general, we will be concerned with specifications of the form:

$$\begin{aligned} y_t &= x_t'\beta + u_t \\ u_t &= z_{t-1}'\gamma + \epsilon_t \end{aligned} \tag{17.1}$$

where x_t is a vector of explanatory variables observed at time t, z_{t-1} is a vector of variables known in the previous period, β and γ are vectors of parameters, u_t is a disturbance term, and ϵ_t is the innovation in the disturbance. The vector z_{t-1} may contain lagged values of u, lagged values of ϵ, or both.

The disturbance u_t is termed the *unconditional residual*. It is the residual based on the structural component ($x_t'\beta$) but not using the information contained in z_{t-1}. The innovation ϵ_t is also known as the *one-period ahead forecast error* or the *prediction error*. It is the difference between the actual value of the dependent variable and a forecast made on the basis of the independent variables and the past forecast errors.

The First-Order Autoregressive Model

The simplest and most widely used model of serial correlation is the first-order autoregressive, or AR(1), model. The AR(1) model is specified as:

$$\begin{aligned} y_t &= x_t'\beta + u_t \\ u_t &= \rho u_{t-1} + \epsilon_t \end{aligned} \tag{17.2}$$

The parameter ρ is the first-order serial correlation coefficient. In effect, the AR(1) model incorporates the residual from the past observation into the regression model for the current observation.

Higher-Order Autoregressive Models

More generally, a regression with an autoregressive process of order p, AR(p) error is given by:

$$\begin{aligned} y_t &= x_t'\beta + u_t \\ u_t &= \rho_1 u_{t-1} + \rho_2 u_{t-2} + \ldots + \rho_p u_{t-p} + \epsilon_t \end{aligned} \tag{17.3}$$

The autocorrelations of a stationary AR(p) process gradually die out to zero, while the partial autocorrelations for lags larger than p are zero.

Testing for Serial Correlation

Before you use an estimated equation for statistical inference (*e.g.* hypothesis tests and forecasting), you should generally examine the residuals for evidence of serial correlation. EViews provides several methods of testing a specification for the presence of serial correlation.

The Durbin-Watson Statistic

EViews reports the Durbin-Watson (DW) statistic as a part of the standard regression output. The Durbin-Watson statistic is a test for first-order serial correlation. More formally, the DW statistic measures the linear association between adjacent residuals from a regression model. The Durbin-Watson is a test of the hypothesis $\rho = 0$ in the specification:

$$u_t = \rho u_{t-1} + \epsilon_t . \tag{17.4}$$

If there is no serial correlation, the DW statistic will be around 2. The DW statistic will fall below 2 if there is positive serial correlation (in the worst case, it will be near zero). If there is negative correlation, the statistic will lie somewhere between 2 and 4.

Positive serial correlation is the most commonly observed form of dependence. As a rule of thumb, with 50 or more observations and only a few independent variables, a DW statistic below about 1.5 is a strong indication of positive first order serial correlation. See Johnston and DiNardo (1997, Chapter 6.6.1) for a thorough discussion on the Durbin-Watson test and a table of the significance points of the statistic.

There are three main limitations of the DW test as a test for serial correlation. First, the distribution of the DW statistic under the null hypothesis depends on the data matrix x . The usual approach to handling this problem is to place bounds on the critical region, creating a region where the test results are inconclusive. Second, if there are lagged dependent variables on the right-hand side of the regression, the DW test is no longer valid. Lastly, you may only test the null hypothesis of no serial correlation against the alternative hypothesis of first-order serial correlation.

Two other tests of serial correlation—the Q-statistic and the Breusch-Godfrey LM test—overcome these limitations, and are preferred in most applications.

Correlograms and Q-statistics

If you select **View/Residual Tests/Correlogram-Q-statistics** on the equation toolbar, EViews will display the autocorrelation and partial autocorrelation functions of the residuals, together with the Ljung-Box Q-statistics for high-order serial correlation. If there is no serial correlation in the residuals, the autocorrelations and partial autocorrelations at all lags should be nearly zero, and all Q-statistics should be insignificant with large p-values.

Note that the p-values of the Q-statistics will be computed with the degrees of freedom adjusted for the inclusion of ARMA terms in your regression. There is evidence that some care should be taken in interpreting the results of a Ljung-Box test applied to the residuals from an ARMAX specification (see Dezhbaksh, 1990, for simulation evidence on the finite sample performance of the test in this setting).

Details on the computation of correlograms and Q-statistics are provided in greater detail in Chapter 11, “Series”, on page 328.

Serial Correlation LM Test

Selecting **View/Residual Tests/Serial Correlation LM Test…** carries out the Breusch-Godfrey Lagrange multiplier test for general, high-order, ARMA errors. In the Lag Specification dialog box, you should enter the highest order of serial correlation to be tested.

The null hypothesis of the test is that there is no serial correlation in the residuals up to the specified order. EViews reports a statistic labeled “F-statistic” and an “Obs*R-squared”

(NR^2—the number of observations times the R-square) statistic. The NR^2 statistic has an asymptotic χ^2 distribution under the null hypothesis. The distribution of the F-statistic is not known, but is often used to conduct an informal test of the null.

See "Serial Correlation LM Test" on page 581 for further discussion of the serial correlation LM test.

Example

As an example of the application of these testing procedures, consider the following results from estimating a simple consumption function by ordinary least squares:

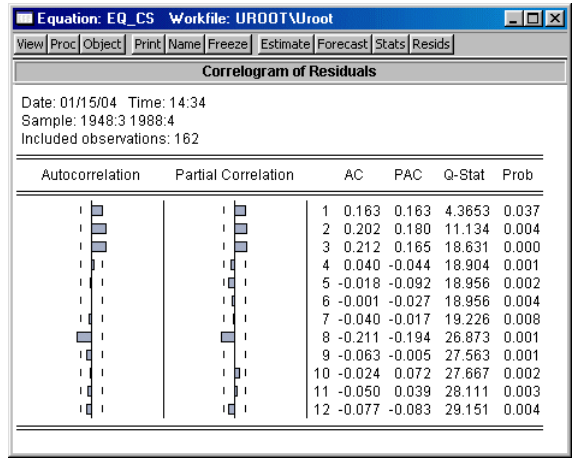
Dependent Variable: CS
Method: Least Squares
Date: 08/19/97 Time: 13:03
Sample: 1948:3 1988:4
Included observations: 162

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-9.227624	5.898177	-1.564487	0.1197
GDP	0.038732	0.017205	2.251193	0.0257
CS(-1)	0.952049	0.024484	38.88516	0.0000

R-squared	0.999625	Mean dependent var	1781.675
Adjusted R-squared	0.999621	S.D. dependent var	694.5419
S.E. of regression	13.53003	Akaike info criterion	8.066045
Sum squared resid	29106.82	Schwarz criterion	8.123223
Log likelihood	-650.3497	F-statistic	212047.1
Durbin-Watson stat	1.672255	Prob(F-statistic)	0.000000

A quick glance at the results reveals that the coefficients are statistically significant and the fit is very tight. However, if the error term is serially correlated, the estimated OLS standard errors are invalid and the estimated coefficients will be biased and inconsistent due to the presence of a lagged dependent variable on the right-hand side. The Durbin-Watson statistic is not appropriate as a test for serial correlation in this case, since there is a lagged dependent variable on the right-hand side of the equation.

Selecting **View/Residual Tests/Correlogram-Q-statistics** for the first 12 lags from this equation produces the following view:



The correlogram has spikes at lags up to three and at lag eight. The Q-statistics are significant at all lags, indicating significant serial correlation in the residuals.

Selecting **View/Residual Tests/Serial Correlation LM Test…** and entering a lag of 4 yields the following result:

Breusch-Godfrey Serial Correlation LM Test:			
F-statistic	3.654696	Probability	0.007109
Obs*R-squared	13.96215	Probability	0.007417

The test rejects the hypothesis of no serial correlation up to order four. The Q-statistic and the LM test both indicate that the residuals are serially correlated and the equation should be re-specified before using it for hypothesis tests and forecasting.

Estimating AR Models

Before you use the tools described in this section, you may first wish to examine your model for other signs of misspecification. Serial correlation in the errors may be evidence of serious problems with your specification. In particular, you should be on guard for an excessively restrictive specification that you arrived at by experimenting with ordinary least squares. Sometimes, adding improperly excluded variables to your regression will eliminate the serial correlation.

For a discussion of the efficiency gains from the serial correlation correction and some Monte-Carlo evidence, see Rao and Griliches (l969).

First-Order Serial Correlation

To estimate an AR(1) model in EViews, open an equation by selecting **Quick/Estimate Equation...** and enter your specification as usual, adding the special expression "AR(1)" to the end of your list. For example, to estimate a simple consumption function with AR(1) errors,

$$\begin{aligned} CS_t &= c_1 + c_2 GDP_t + u_t \\ u_t &= \rho u_{t-1} + \epsilon_t \end{aligned} \tag{17.5}$$

you should specify your equation as:

```
cs c gdp ar(1)
```

EViews automatically adjusts your sample to account for the lagged data used in estimation, estimates the model, and reports the adjusted sample along with the remainder of the estimation output.

Higher-Order Serial Correlation

Estimating higher order AR models is only slightly more complicated. To estimate an AR(k), you should enter your specification, followed by expressions for each AR term you wish to include. If you wish to estimate a model with autocorrelations from one to five:

$$\begin{aligned} CS_t &= c_1 + c_2 GDP_t + u_t \\ u_t &= \rho_1 u_{t-1} + \rho_2 u_{t-2} + \ldots + \rho_5 u_{t-5} + \epsilon_t \end{aligned} \tag{17.6}$$

you should enter:

```
cs c gdp ar(1) ar(2) ar(3) ar(4) ar(5)
```

By requiring that you enter all of the autocorrelations you wish to include in your model, EViews allows you great flexibility in restricting lower order correlations to be zero. For example, if you have quarterly data and want to include a single term to account for seasonal autocorrelation, you could enter

```
cs c gdp ar(4)
```

Nonlinear Models with Serial Correlation

EViews can estimate nonlinear regression models with additive AR errors. For example, suppose you wish to estimate the following nonlinear specification with an AR(2) error:

$$\begin{aligned} CS_t &= c_1 + GDP_t^{c_2} + u_t \\ u_t &= c_3 u_{t-1} + c_4 u_{t-2} + \epsilon_t \end{aligned} \tag{17.7}$$

Simply specify your model using EViews expressions, followed by an additive term describing the AR correction enclosed in square brackets. The AR term should contain a coefficient assignment for each AR lag, separated by commas:

```
cs = c(1) + gdp^c(2) + [ar(1)=c(3), ar(2)=c(4)]
```

EViews transforms this nonlinear model by differencing, and estimates the transformed nonlinear specification using a Gauss-Newton iterative procedure (see "How EViews Estimates AR Models" on page 500).

Two-Stage Regression Models with Serial Correlation

By combining two-stage least squares or two-stage nonlinear least squares with AR terms, you can estimate models where there is correlation between regressors and the innovations as well as serial correlation in the residuals.

If the original regression model is linear, EViews uses the Marquardt algorithm to estimate the parameters of the transformed specification. If the original model is nonlinear, EViews uses Gauss-Newton to estimate the AR corrected specification.

For further details on the algorithms and related issues associated with the choice of instruments, see the discussion in "TSLS with AR errors" beginning on page 477.

Output from AR Estimation

When estimating an AR model, some care must be taken in interpreting your results. While the estimated coefficients, coefficient standard errors, and t-statistics may be interpreted in the usual manner, results involving residuals differ from those computed in OLS settings.

To understand these differences, keep in mind that there are two different residuals associated with an AR model. The first are the estimated *unconditional residuals*:

$$\hat{u}_t = y_t - x_t' b \,, \tag{17.8}$$

which are computed using the original variables, and the estimated coefficients, b. These residuals are the errors that you would observe if you made a prediction of the value of y_t using contemporaneous information, but ignoring the information contained in the lagged residual.

Normally, there is no strong reason to examine these residuals, and EViews does not automatically compute them following estimation.

The second set of residuals are the estimated *one-period ahead forecast errors*, $\hat{\epsilon}$. As the name suggests, these residuals represent the forecast errors you would make if you computed forecasts using a prediction of the residuals based upon past values of your data, in addition to the contemporaneous information. In essence, you improve upon the uncondi-

tional forecasts and residuals by taking advantage of the predictive power of the lagged residuals.

For AR models, the residual-based regression statistics—such as the R^2, the standard error of regression, and the Durbin-Watson statistic— reported by EViews are based on the one-period ahead forecast errors, $\hat{\epsilon}$.

A set of statistics that is unique to AR models is the estimated AR parameters, $\hat{\rho}_i$. For the simple AR(1) model, the estimated parameter $\hat{\rho}$ is the serial correlation coefficient of the unconditional residuals. For a stationary AR(1) model, the true ρ lies between –1 (extreme negative serial correlation) and +1 (extreme positive serial correlation). The stationarity condition for general AR(p) processes is that the inverted roots of the lag polynomial lie inside the unit circle. EViews reports these roots as **Inverted AR Roots** at the bottom of the regression output. There is no particular problem if the roots are imaginary, but a stationary AR model should have all roots with modulus less than one.

How EViews Estimates AR Models

Textbooks often describe techniques for estimating AR models. The most widely discussed approaches, the Cochrane-Orcutt, Prais-Winsten, Hatanaka, and Hildreth-Lu procedures, are multi-step approaches designed so that estimation can be performed using standard *linear* regression. All of these approaches suffer from important drawbacks which occur when working with models containing lagged dependent variables as regressors, or models using higher-order AR specifications; see Davidson and MacKinnon (1994, pp. 329–341), Greene (1997, p. 600–607).

EViews estimates AR models using nonlinear regression techniques. This approach has the advantage of being easy to understand, generally applicable, and easily extended to nonlinear specifications and models that contain endogenous right-hand side variables. Note that the nonlinear least squares estimates are asymptotically equivalent to maximum likelihood estimates and are asymptotically efficient.

To estimate an AR(1) model, EViews transforms the linear model,

$$\begin{aligned} y_t &= x_t'\beta + u_t \\ u_t &= \rho u_{t-1} + \epsilon_t \end{aligned} \tag{17.9}$$

into the nonlinear model:

$$y_t = \rho y_{t-1} + (x_t - \rho x_{t-1})'\beta + \epsilon_t, \tag{17.10}$$

by substituting the second equation into the first, and rearranging terms. The coefficients ρ and β are estimated simultaneously by applying a Marquardt nonlinear least squares algorithm to the transformed equation. See Appendix C, “Estimation and Solution Options”, on page 951 for details on nonlinear estimation.

For a nonlinear AR(1) specification, EViews transforms the nonlinear model,

$$\begin{aligned} y_t &= f(x_t, \beta) + u_t \\ u_t &= \rho u_{t-1} + \epsilon_t \end{aligned} \tag{17.11}$$

into the alternative nonlinear specification:

$$y_t = \rho y_{t-1} + f(x_t, \beta) - \rho f(x_{t-1}, \beta) + \epsilon_t \tag{17.12}$$

and estimates the coefficients using a Marquardt nonlinear least squares algorithm.

Higher order AR specifications are handled analogously. For example, a nonlinear AR(3) is estimated using nonlinear least squares on the equation:

$$\begin{aligned} y_t = (\rho_1 y_{t-1} + \rho_2 y_{t-2} + \rho_3 y_{t-3}) + f(x_t, \beta) - \rho_1 f(x_{t-1}, \beta) \\ - \rho_2 f(x_{t-2}, \beta) - \rho_3 f(x_{t-3}, \beta) + \epsilon_t \end{aligned} \tag{17.13}$$

For details, see Fair (1984, pp. 210–214), and Davidson and MacKinnon (1996, pp. 331–341).

ARIMA Theory

ARIMA (autoregressive integrated moving average) models are generalizations of the simple AR model that use three tools for modeling the serial correlation in the disturbance:

- The first tool is the autoregressive, or AR, term. The AR(1) model introduced above uses only the first-order term, but in general, you may use additional, higher-order AR terms. Each AR term corresponds to the use of a lagged value of the residual in the forecasting equation for the unconditional residual. An autoregressive model of order p, AR(p) has the form

$$u_t = \rho_1 u_{t-1} + \rho_2 u_{t-2} + \ldots + \rho_p u_{t-p} + \epsilon_t . \tag{17.14}$$

- The second tool is the integration order term. Each integration order corresponds to differencing the series being forecast. A first-order integrated component means that the forecasting model is designed for the first difference of the original series. A second-order component corresponds to using second differences, and so on.

- The third tool is the MA, or moving average term. A moving average forecasting model uses lagged values of the forecast error to improve the current forecast. A first-order moving average term uses the most recent forecast error, a second-order term uses the forecast error from the two most recent periods, and so on. An MA(q) has the form:

$$u_t = \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q} . \tag{17.15}$$

> Please be aware that some authors and software packages use the opposite sign convention for the θ coefficients so that the signs of the MA coefficients may be reversed.

The autoregressive and moving average specifications can be combined to form an ARMA(p, q) specification:

$$u_t = \rho_1 u_{t-1} + \rho_2 u_{t-2} + \ldots + \rho_p u_{t-p} + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \ldots + \theta_q \epsilon_{t-q} \quad (17.16)$$

Although econometricians typically use ARIMA models applied to the residuals from a regression model, the specification can also be applied directly to a series. This latter approach provides a univariate model, specifying the conditional mean of the series as a constant, and measuring the residuals as differences of the series from its mean.

Principles of ARIMA Modeling (Box-Jenkins 1976)

In ARIMA forecasting, you assemble a complete forecasting model by using combinations of the three building blocks described above. The first step in forming an ARIMA model for a series of residuals is to look at its autocorrelation properties. You can use the correlogram view of a series for this purpose, as outlined in "Correlogram" on page 326.

This phase of the ARIMA modeling procedure is called *identification* (not to be confused with the same term used in the simultaneous equations literature). The nature of the correlation between current values of residuals and their past values provides guidance in selecting an ARIMA specification.

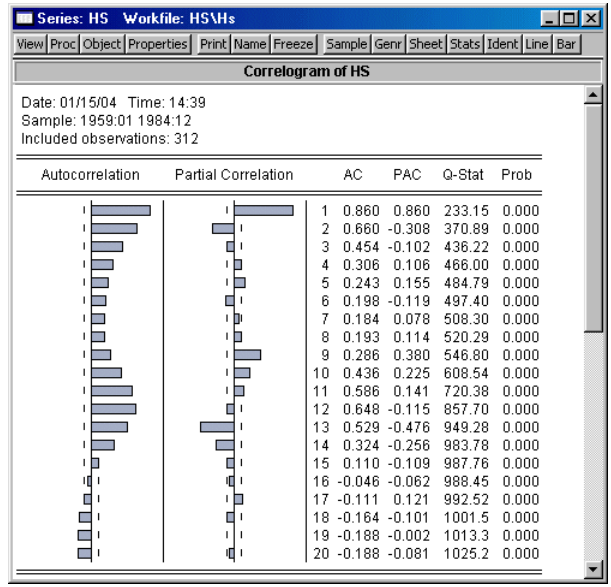
The autocorrelations are easy to interpret—each one is the correlation coefficient of the current value of the series with the series lagged a certain number of periods. The partial autocorrelations are a bit more complicated; they measure the correlation of the current and lagged series after taking into account the predictive power of all the values of the series with smaller lags. The partial autocorrelation for lag 6, for example, measures the added predictive power of u_{t-6} when $u_1, \ldots, u_{t-5}$ are already in the prediction model. In fact, the partial autocorrelation is precisely the regression coefficient of u_{t-6} in a regression where the earlier lags are also used as predictors of u_t.

If you suspect that there is a distributed lag relationship between your dependent (left-hand) variable and some other predictor, you may want to look at their cross correlations before carrying out estimation.

The next step is to decide what kind of ARIMA model to use. If the autocorrelation function dies off smoothly at a geometric rate, and the partial autocorrelations were zero after one lag, then a first-order autoregressive model is appropriate. Alternatively, if the autocorrelations were zero after one lag and the partial autocorrelations declined geometrically, a first-order moving average process would seem appropriate. If the autocorrelations appear

to have a seasonal pattern, this would suggest the presence of a seasonal ARMA structure (see "Seasonal ARMA Terms" on page 506).

For example, we can examine the correlogram of the DRI Basics housing series in the HS.WF1 workfile by selecting **View/Correlogram...** from the HS series toolbar:



The "wavy" cyclical correlogram with a seasonal frequency suggests fitting a seasonal ARMA model to HS.

The goal of ARIMA analysis is a parsimonious representation of the process governing the residual. You should use only enough AR and MA terms to fit the properties of the residuals. The Akaike information criterion and Schwarz criterion provided with each set of estimates may also be used as a guide for the appropriate lag order selection.

After fitting a candidate ARIMA specification, you should verify that there are no remaining autocorrelations that your model has not accounted for. Examine the autocorrelations and the partial autocorrelations of the innovations (the residuals from the ARIMA model) to see if any important forecasting power has been overlooked. EViews provides views for diagnostic checks after estimation.

Estimating ARIMA Models

EViews estimates general ARIMA specifications that allow for right-hand side explanatory variables. Despite the fact that these models are sometimes termed ARIMAX specifications, we will refer to this general class of models as ARIMA.

To specify your ARIMA model, you will:

- Difference your dependent variable, if necessary, to account for the order of integration.
- Describe your structural regression model (dependent variables and regressors) and add any AR or MA terms, as described above.

Differencing

The `d` operator can be used to specify differences of series. To specify first differencing, simply include the series name in parentheses after `d`. For example, `d(gdp)` specifies the first difference of GDP, or GDP–GDP(–1).

More complicated forms of differencing may be specified with two optional parameters, n and s. `d(x,n)` specifies the n-th order difference of the series X:

$$d(x, n) = (1 - L)^n x , \tag{17.17}$$

where L is the lag operator. For example, `d(gdp,2)` specifies the second order difference of GDP:

```
d(gdp,2) = gdp - 2*gdp(-1) + gdp(-2)
```

`d(x,n,s)` specifies n-th order ordinary differencing of X with a seasonal difference at lag s:

$$d(x, n, s) = (1 - L)^n (1 - L^s) x . \tag{17.18}$$

For example, `d(gdp,0,4)` specifies zero ordinary differencing with a seasonal difference at lag 4, or GDP–GDP(–4).

If you need to work in logs, you can also use the `dlog` operator, which returns differences in the log values. For example, `dlog(gdp)` specifies the first difference of log(GDP) or log(GDP)–log(GDP(–1)). You may also specify the n and s options as described for the simple `d` operator, `dlog(x,n,s)`.

There are two ways to estimate integrated models in EViews. First, you may generate a new series containing the differenced data, and then estimate an ARMA model using the new data. For example, to estimate a Box-Jenkins ARIMA(1, 1, 1) model for M1, you can enter:

```
series dm1 = d(m1)
equation eq1.ls dm1 c ar(1) ma(1)
```

Alternatively, you may include the difference operator `d` directly in the estimation specification. For example, the same ARIMA(1,1,1) model can be estimated using the command:

```
equation eq1.ls d(m1) c ar(1) ma(1)
```

The latter method should generally be preferred for an important reason. If you define a new variable, such as DM1 above, and use it in your estimation procedure, then when you forecast from the estimated model, EViews will make forecasts of the dependent variable DM1. That is, you will get a forecast of the differenced series. If you are really interested in forecasts of the level variable, in this case M1, you will have to manually transform the forecasted value and adjust the computed standard errors accordingly. Moreover, if any

other transformation or lags of M1 are included as regressors, EViews will not know that they are related to DM1. If, however, you specify the model using the difference operator expression for the dependent variable, `d(m1)`, the forecasting procedure will provide you with the option of forecasting the level variable, in this case M1.

The difference operator may also be used in specifying exogenous variables and can be used in equations without ARMA terms. Simply include them in the list of regressors in addition to the endogenous variables. For example:

```
d(cs,2) c d(gdp,2) d(gdp(-1),2) d(gdp(-2),2) time
```

is a valid specification that employs the difference operator on both the left-hand and right-hand sides of the equation.

ARMA Terms

The AR and MA parts of your model will be specified using the keywords `ar` and `ma` as part of the equation. We have already seen examples of this approach in our specification of the AR terms above, and the concepts carry over directly to MA terms.

For example, to estimate a second-order autoregressive and first-order moving average error process ARMA(2,1), you would include expressions for the AR(1), AR(2), and MA(1) terms along with your other regressors:

```
c gov ar(1) ar(2) ma(1)
```

Once again, you need not use the AR and MA terms consecutively. For example, if you want to fit a fourth-order autoregressive model to take account of seasonal movements, you could use AR(4) by itself:

```
c gov ar(4)
```

You may also specify a pure moving average model by using only MA terms. Thus:

```
c gov ma(1) ma(2)
```

indicates an MA(2) model for the residuals.

The traditional Box-Jenkins or ARMA models do not have any right-hand side variables except for the constant. In this case, your list of regressors would just contain a C in addition to the AR and MA terms. For example:

```
c ar(1) ar(2) ma(1) ma(2)
```

is a standard Box-Jenkins ARMA (2,2).

Seasonal ARMA Terms

Box and Jenkins (1976) recommend the use of seasonal autoregressive (SAR) and seasonal moving average (SMA) terms for monthly or quarterly data with systematic seasonal movements. A SAR(p) term can be included in your equation specification for a seasonal autoregressive term with lag p. The lag polynomial used in estimation is the product of the one specified by the AR terms and the one specified by the SAR terms. The purpose of the SAR is to allow you to form the product of lag polynomials.

Similarly, SMA(q) can be included in your specification to specify a seasonal moving average term with lag q. The lag polynomial used in estimation is the product of the one defined by the MA terms and the one specified by the SMA terms. As with the SAR, the SMA term allows you to build up a polynomial that is the product of underlying lag polynomials.

For example, a second-order AR process without seasonality is given by,

$$u_t = \rho_1 u_{t-1} + \rho_2 u_{t-2} + \epsilon_t, \tag{17.19}$$

which can be represented using the lag operator L, $L^n x_t = x_{t-n}$ as:

$$(1 - \rho_1 L - \rho_2 L^2) u_t = \epsilon_t. \tag{17.20}$$

You can estimate this process by including `ar(1)` and `ar(2)` terms in the list of regressors. With quarterly data, you might want to add a `sar(4)` expression to take account of seasonality. If you specify the equation as,

```
sales c inc ar(1) ar(2) sar(4)
```

then the estimated error structure would be:

$$(1 - \rho_1 L - \rho_2 L^2)(1 - \theta L^4) u_t = \epsilon_t. \tag{17.21}$$

The error process is equivalent to:

$$u_t = \rho_1 u_{t-1} + \rho_2 u_{t-2} + \theta u_{t-4} - \theta\rho_1 u_{t-5} - \theta\rho_2 u_{t-6} + \epsilon_t. \tag{17.22}$$

The parameter θ is associated with the seasonal part of the process. Note that this is an AR(6) process with nonlinear restrictions on the coefficients.

As another example, a second-order MA process without seasonality may be written,

$$u_t = \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2}, \tag{17.23}$$

or using lag operators:

$$u_t = (1 + \theta_1 L + \theta_2 L^2) \epsilon_t. \tag{17.24}$$

You may estimate this second-order process by including both the MA(1) and MA(2) terms in your equation specification.

With quarterly data, you might want to add `sma(4)` to take account of seasonality. If you specify the equation as,

```
cs c ad ma(1) ma(2) sma(4)
```

then the estimated model is:

$$\begin{aligned} CS_t &= \beta_1 + \beta_2 AD_t + u_t \\ u_t &= (1 + \theta_1 L + \theta_2 L^2)(1 + \omega L^4)\epsilon_t \end{aligned} \tag{17.25}$$

The error process is equivalent to:

$$u_t = \epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2} + \omega\epsilon_{t-4} + \omega\theta_1\epsilon_{t-5} + \omega\theta_2\epsilon_{t-6}\,. \tag{17.26}$$

The parameter w is associated with the seasonal part of the process. This is just an MA(6) process with nonlinear restrictions on the coefficients. You can also include both SAR and SMA terms.

Output from ARIMA Estimation

The output from estimation with AR or MA specifications is the same as for ordinary least squares, with the addition of a lower block that shows the reciprocal roots of the AR and MA polynomials. If we write the general ARMA model using the lag polynomial $\rho(L)$ and $\theta(L)$ as,

$$\rho(L)u_t = \theta(L)\epsilon_t\,, \tag{17.27}$$

then the reported roots are the roots of the polynomials:

$$\rho(x^{-1}) = 0 \qquad \text{and} \qquad \theta(x^{-1}) = 0\,. \tag{17.28}$$

The roots, which may be imaginary, should have modulus no greater than one. The output will display a warning message if any of the roots violate this condition.

If ρ has a real root whose absolute value exceeds one or a pair of complex reciprocal roots outside the unit circle (that is, with modulus greater than one), it means that the autoregressive process is explosive.

If θ has reciprocal roots outside the unit circle, we say that the MA process is *noninvertible*, which makes interpreting and using the MA results difficult. However, noninvertibility poses no substantive problem, since as Hamilton (1994a, p. 65) notes, there is always an equivalent representation for the MA model where the reciprocal roots lie inside the unit circle. Accordingly, you should re-estimate your model with different starting values until you get a moving average process that satisfies invertibility. Alternatively, you may wish to turn off MA backcasting (see "Backcasting MA terms" on page 510).

If the estimated MA process has roots with modulus close to one, it is a sign that you may have over-differenced the data. The process will be difficult to estimate and even more difficult to forecast. If possible, you should re-estimate with one less round of differencing.

Consider the following example output from ARMA estimation:

Dependent Variable: R
Method: Least Squares
Date: 01/15/04 Time: 14:45
Sample (adjusted): 1954:06 1993:07
Included observations: 470 after adjusting endpoints
Convergence achieved after 24 iterations
Backcast: 1954:01 1954:05

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	8.638235	1.201220	7.191220	0.0000
AR(1)	0.982695	0.011021	89.16753	0.0000
SAR(4)	0.965504	0.017051	56.62427	0.0000
MA(1)	0.511001	0.040194	12.71339	0.0000
SMA(4)	-0.979706	0.009678	-101.2314	0.0000

R-squared	0.991576	Mean dependent var	6.978830
Adjusted R-squared	0.991504	S.D. dependent var	2.919607
S.E. of regression	0.269115	Akaike info criterion	0.223228
Sum squared resid	33.67674	Schwarz criterion	0.267406
Log likelihood	-47.45869	F-statistic	13683.92
Durbin-Watson stat	2.100675	Prob(F-statistic)	0.000000

Inverted AR Roots	.99	.98		
Inverted MA Roots	.99	.00+.99i	-.00-.99i	-.51
	-.99			

This estimation result corresponds to the following specification,

$$\begin{aligned} y_t &= 8.64 + u_t \\ (1-0.98L)(1-0.97L^4)u_t &= (1+0.51L)(1-0.98L^4)\epsilon_t \end{aligned} \tag{17.29}$$

or equivalently, to:

$$\begin{aligned} y_t = {} & 0.0052 + 0.98y_{t-1} + 0.97y_{t-4} - 0.95y_{t-5} + \epsilon_t \\ & + 0.51\epsilon_{t-1} - 0.98\epsilon_{t-4} - 0.50\epsilon_{t-5} \end{aligned} \tag{17.30}$$

Note that the signs of the MA terms may be reversed from those in textbooks. Note also that the inverted roots have moduli very close to one, which is typical for many macro time series models.

Estimation Options

ARMA estimation employs the same nonlinear estimation techniques described earlier for AR estimation. These nonlinear estimation techniques are discussed further in Chapter 16, "Additional Regression Methods", on page 481.

You may need to use the **Estimation Options** dialog box to control the iterative process. EViews provides a number of options that allow you to control the iterative procedure of the estimation algorithm. In general, you can rely on the EViews choices, but on occasion you may wish to override the default settings.

Iteration Limits and Convergence Criterion

Controlling the maximum number of iterations and convergence criterion are described in detail in "Iteration and Convergence Options" on page 953.

Derivative Methods

EViews always computes the derivatives of AR coefficients analytically and the derivatives of the MA coefficients using finite difference numeric derivative methods. For other coefficients in the model, EViews provides you with the option of computing analytic expressions for derivatives of the regression equation (if possible) or computing finite difference numeric derivatives in cases where the derivative is not constant. Furthermore, you can choose whether to favor speed of computation (fewer function evaluations) or whether to favor accuracy (more function evaluations) in the numeric derivative computation.

Starting Values for ARMA Estimation

As discussed above, models with AR or MA terms are estimated by nonlinear least squares. Nonlinear estimation techniques require starting values for all coefficient estimates. Normally, EViews determines its own starting values and for the most part this is an issue that you need not be concerned about. However, there are a few times when you may want to override the default starting values.

First, estimation will sometimes halt when the maximum number of iterations is reached, despite the fact that convergence is not achieved. Resuming the estimation with starting values from the previous step causes estimation to pick up where it left off instead of starting over. You may also want to try different starting values to ensure that the estimates are a global rather than a local minimum of the squared errors. You might also want to supply starting values if you have a good idea of what the answers should be, and want to speed up the estimation process.

To control the starting values for ARMA estimation, click on the **Options** button in the Equation Specification dialog. Among the options which EViews provides are several alternatives for setting starting values that you can see by accessing the drop-down menu labeled **Starting Coefficient Values for ARMA**.

EViews' default approach is **OLS/TSLS**, which runs a preliminary estimation without the ARMA terms and then starts nonlinear estimation from those values. An alternative is to use fractions of the OLS or TSLS coefficients as starting values. You can choose **.8**, **.5**, **.3**, or you can start with all coefficient values set equal to zero.

The final starting value option is **User Supplied**. Under this option, EViews uses the coefficient values that are in the coefficient vector. To set the starting values, open a window for the coefficient vector C by double clicking on the icon, and editing the values.

To properly set starting values, you will need a little more information about how EViews assigns coefficients for the ARMA terms. As with other estimation methods, when you specify your equation as a list of variables, EViews uses the built-in C coefficient vector. It assigns coefficient numbers to the variables in the following order:

- First are the coefficients of the variables, in order of entry.
- Next come the AR terms in the order you typed them.
- The SAR, MA, and SMA coefficients follow, in that order.

Thus the following two specifications will have their coefficients in the same order:

```
y c x ma(2) ma(1) sma(4) ar(1)
y sma(4)c ar(1) ma(2) x ma(1)
```

You may also assign values in the C vector using the `param` command:

```
param c(1) 50 c(2) .8 c(3) .2 c(4) .6 c(5) .1 c(6) .5
```

The starting values will be 50 for the constant, 0.8 for X, 0.2 for AR(1), 0.6 for MA(2), 0.1 for MA(1) and 0.5 for SMA(4). Following estimation, you can always see the assignment of coefficients by looking at the **Representations** view of your equation.

You can also fill the C vector from any estimated equation (without typing the numbers) by choosing **Proc/Update Coefs from Equation** in the equation toolbar.

Backcasting MA terms

By default, EViews backcasts MA terms (Box and Jenkins, 1976). Consider an MA(q) model of the form:

$$\begin{aligned} y_t &= X_t'\beta + u_t \\ u_t &= \epsilon_t + \theta_1\epsilon_{t-1} + \theta_2\epsilon_{t-2} + \ldots + \theta_q\epsilon_{t-q} \end{aligned} \tag{17.31}$$

Given initial values, $\hat{\beta}$ and $\hat{\theta}$, EViews first computes the unconditional residuals $\hat{u}_t$ for $t = 1, 2, \ldots, T$, and uses the backward recursion:

$$\tilde{\epsilon}_t = \hat{u}_t - \hat{\theta}_1\tilde{\epsilon}_{t+1} - \ldots - \hat{\theta}_q\tilde{\epsilon}_{t+q} \tag{17.32}$$

to compute backcast values of ϵ to $\epsilon_{-(q-1)}$. To start this recursion, the q values for the innovations *beyond* the estimation sample are set to zero:

$$\tilde{\epsilon}_{T+1} = \tilde{\epsilon}_{T+2} = \ldots = \tilde{\epsilon}_{T+q} = 0\,. \tag{17.33}$$

Next, a forward recursion is used to estimate the values of the innovations:

$$\hat{\epsilon}_t = \hat{u}_t - \hat{\theta}_1\hat{\epsilon}_{t-1} - \ldots - \hat{\theta}_q\hat{\epsilon}_{t-q}, \tag{17.34}$$

using the backcasted values of the innovations (to initialize the recursion) and the actual residuals. If your model also includes AR terms, EViews will ρ-difference the $\hat{u}_t$ to eliminate the serial correlation prior to performing the backcast.

Lastly, the sum of squared residuals (SSR) is formed as a function of the β and θ, using the fitted values of the lagged innovations:

$$\text{ssr}(\beta, \theta) = \sum_{t=p+1}^{T} (y_t - X_t'\beta - \theta_1\hat{\epsilon}_{t-1} - \ldots - \theta_q\hat{\epsilon}_{t-q})^2\,. \tag{17.35}$$

This expression is minimized with respect to β and θ.

The backcast step, forward recursion, and minimization procedures are repeated until the estimates of β and θ converge.

If backcasting is turned off, the values of the pre-sample ϵ are set to zero:

$$\epsilon_{-(q-1)} = \ldots = \hat{\epsilon}_0 = 0\,, \tag{17.36}$$

and forward recursion is used to solve for the remaining values of the innovations.

Dealing with Estimation Problems

Since EViews uses nonlinear least squares algorithms to estimate ARMA models, all of the discussion in Chapter 16, "Solving Estimation Problems" on page 487, is applicable, especially the advice to try alternative starting values.

There are a few other issues to consider that are specific to estimation of ARMA models.

First, MA models are notoriously difficult to estimate. In particular, you should avoid high order MA terms unless absolutely required for your model as they are likely to cause estimation difficulties. For example, a single large spike at lag 57 in the correlogram does not necessarily require you to include an MA(57) term in your model unless you know there is something special happening every 57 periods. It is more likely that the spike in the correlogram is simply the product of one or more outliers in the series. By including many MA terms in your model, you lose degrees of freedom, and may sacrifice stability and reliability of your estimates.

If the underlying roots of the MA process have modulus close to one, you may encounter estimation difficulties, with EViews reporting that it cannot improve the sum-of-squares or that it failed to converge in the maximum number of iterations. This behavior may be a sign that you have over-differenced the data. You should check the correlogram of the series to determine whether you can re-estimate with one less round of differencing.

Lastly, if you continue to have problems, you may wish to turn off MA backcasting.

TSLS with ARIMA errors

Two-stage least squares or instrumental variable estimation with ARIMA poses no particular difficulties.

For a detailed discussion of how to estimate TSLS specifications with ARMA errors, see "Two-stage Least Squares" on page 473.

Nonlinear Models with ARMA errors

EViews will estimate nonlinear ordinary and two-stage least squares models with autoregressive error terms. For details, see the extended discussion in "Nonlinear Least Squares" beginning on page 480.

EViews does not currently estimate nonlinear models with MA errors. You can, however, use the state space object to specify and estimate these models (see "ARMAX(2, 3) with a Random Coefficient" on page 762).

Weighted Models with ARMA errors

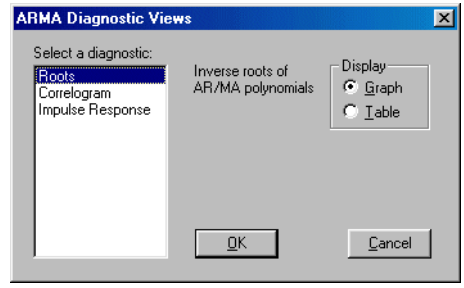
EViews does not have procedures to automatically estimate weighted models with ARMA error terms—if you add AR terms to a weighted model, the weighting series will be ignored. You can, of course, always construct the weighted series and then perform estimation using the weighted data and ARMA terms.

ARMA Equation Diagnostics

ARMA Structure

This set of views provides access to several diagnostic views that help you assess the structure of the ARMA portion of the estimated equation. The view is currently available only for models specified by list that includes at least one AR or MA term and estimated by least squares. There are three views available: roots, correlogram, and impulse response.

To display the ARMA structure, select **View/ ARMA Structure...** from the menu of an estimated equation. If the equation type supports this view and there are no ARMA components in the specification, EViews will open the **ARMA Diagnostic Views** dialog:

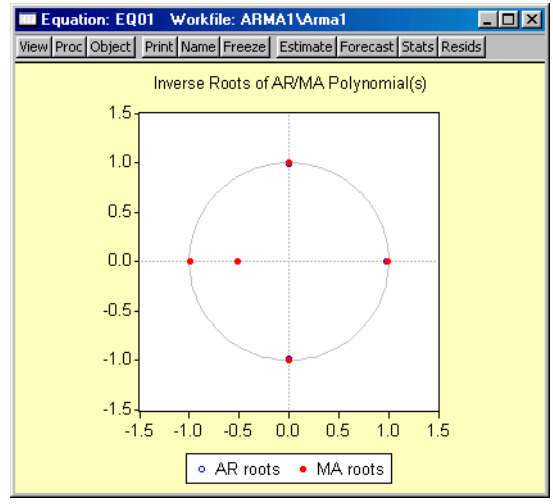


On the left-hand side of the dialog, you will select one of the three types of diagnostics. When you click on one of the types, the right-hand side of the dialog will change to show you the options for each type.

Roots

The roots view displays the inverse roots of the AR and/or MA characteristic polynomial. The roots may be displayed as a graph or as a table by selecting the appropriate radio button.

The graph view plots the roots in the complex plane where the horizontal axis is the real part and the vertical axis is the imaginary part of each root.



If the estimated ARMA process is (covariance) stationary, then all AR roots should lie *inside* the unit circle. If the estimated ARMA process is invertible, then all MA roots should lie *inside* the unit circle. The table view displays all roots in order of decreasing modulus (square root of the sum of squares of the real and imaginary parts).

For imaginary roots (which come in conjugate pairs), we also display the cycle corresponding to that root. The cycle is computed as $2\Pi/a$, where $a = \text{atan}(i/r)$, and i and r are the imaginary and real parts of the root, respectively. The cycle for a real root is infinite and is not reported.

Inverse Roots of AR/MA Polynomial(s)
Specification: R C AR(1) SAR(4) MA(1) SMA(4)
Date: 01/15/04 Time: 14:55
Sample: 1954:01 1994:12
Included observations: 470

AR Root(s)	Modulus	Cycle
-0.991259	0.991259	
8.33e-17 ± 0.991259i	0.991259	4.000000
0.991259	0.991259	
0.982694	0.982694	

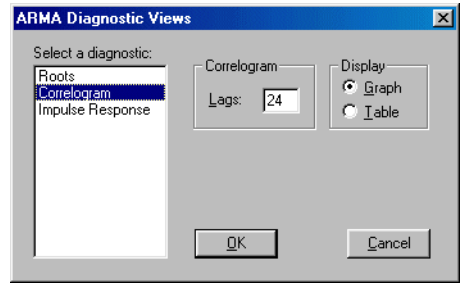
No root lies outside the unit circle.
ARMA model is stationary.

MA Root(s)	Modulus	Cycle
-0.994884	0.994884	
2.17e-16 ± 0.994884i	0.994884	4.000000
0.994884	0.994884	
-0.510994	0.510994	

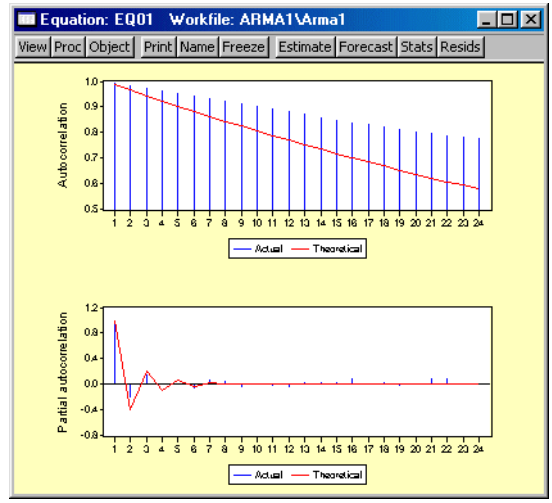
No root lies outside the unit circle.
ARMA model is invertible.

Correlogram

The correlogram view compares the autocorrelation pattern of the structural residuals and that of the estimated model for a specified number of periods (recall that the structural residuals are the residuals after removing the effect of the fitted exogenous regressors but *not* the ARMA terms). For a properly specified model, the residual and theoretical (estimated) autocorrelations and partial autocorrelations should be "close".



To perform the comparison, simply select the **Correlogram** diagnostic, specify a number of lags to be evaluated, and a display format (**Graph** or **Table**).



Here, we have specified a graphical comparison over 24 periods/lags. The graph view plots the autocorrelations and partial autocorrelations of the sample structural residuals and those that are implied from the estimated ARMA parameters. If the estimated ARMA model is not stationary, only the sample second moments from the structural residuals are plotted.

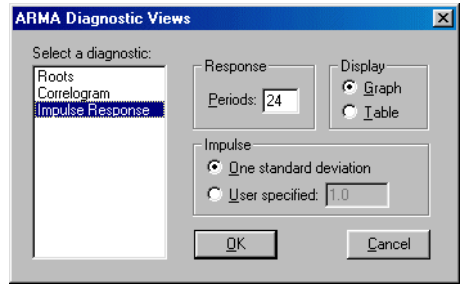
The table view displays the numerical values for each of the second moments and the difference between from the estimated theoretical. If the estimated ARMA model is not stationary, the theoretical second moments implied from the estimated ARMA parameters will be filled with NAs.

Note that the table view starts from lag zero, while the graph view starts from lag one.

Impulse Response

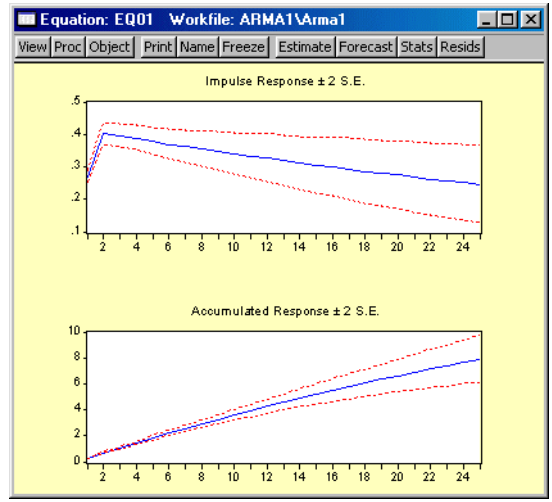
The ARMA impulse response view traces the response of the ARMA part of the estimated equation to shocks in the innovation.

An impulse response function traces the response to a one-time shock in the innovation. The accumulated response is the accumulated sum of the impulse responses. It can be interpreted as the response to step impulse where the same shock occurs in every period from the first.



To compute the impulse response (and accumulated responses), select the **Impulse Response** diagnostic, enter the number of periods, and display type, and define the shock. For the latter, you have the choice of using a one standard deviation shock (using the standard error of the regression for the estimated equation), or providing a user specified value. Note that if you select a one standard deviation shock, EViews will take account of innovation uncertainty when estimating the standard errors of the responses.

If the estimated ARMA model is stationary, the impulse responses will asymptote to zero, while the accumulated responses will asymptote to its long-run value. These asymptotic values will be shown as dotted horizontal lines in the graph view.

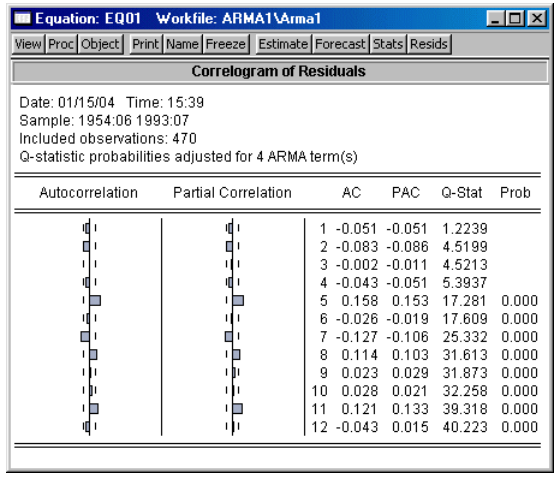


For a highly persistent near unit root but stationary process, the asymptotes may not be drawn in the graph for a short horizon. For a table view, the asymptotic values (together with its standard errors) will be shown at the bottom of the table. If the estimated ARMA process is not stationary, the asymptotic values will not be displayed since they do not exist.

Q-statistics

If your ARMA model is correctly specified, the residuals from the model should be nearly white noise. This means that there should be no serial correlation left in the residuals. The Durbin-Watson statistic reported in the regression output is a test for AR(1) in the absence of lagged dependent variables on the right-hand side. As discussed in "Correlograms and Q-statistics" on page 495, more general tests for serial correlation in the residuals may be carried out with **View/Residual Tests/Correlogram-Q-statistic** and **View/Residual Tests/Serial Correlation LM Test**….

For the example seasonal ARMA model, the 12-period residual correlogram looks as follows:



The correlogram has a significant spike at lag 5, and all subsequent Q-statistics are highly significant. This result clearly indicates the need for respecification of the model.

Nonstationary Time Series

The theory behind ARMA estimation is based on stationary time series. A series is said to be (weakly or covariance) *stationary* if the mean and autocovariances of the series do not depend on time. Any series that is not stationary is said to be *nonstationary*.

A common example of a nonstationary series is the *random walk*:

$$y_t = y_{t-1} + \epsilon_t, \tag{17.37}$$

where ϵ is a stationary random disturbance term. The series y has a constant forecast value, conditional on t, and the variance is increasing over time. The random walk is a difference stationary series since the first difference of y is stationary:

$$y_t - y_{t-1} = (1-L)y_t = \epsilon_t. \tag{17.38}$$

A difference stationary series is said to be *integrated* and is denoted as I(d) where d is the order of integration. The order of integration is the number of unit roots contained in the series, or the number of differencing operations it takes to make the series stationary. For the random walk above, there is one unit root, so it is an I(1) series. Similarly, a stationary series is I(0).

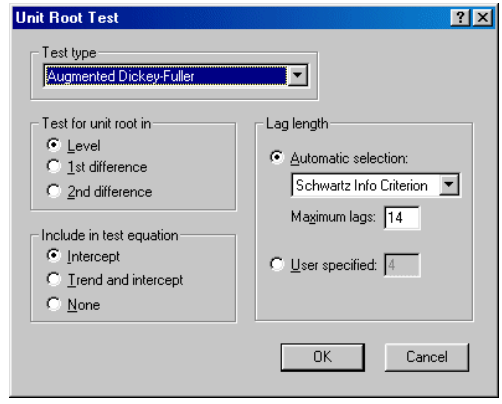
Standard inference procedures do not apply to regressions which contain an integrated dependent variable or integrated regressors. Therefore, it is important to check whether a series is stationary or not before using it in a regression. The formal method to test the stationarity of a series is the unit root test.

Unit Root Tests

EViews provides you with a variety of powerful tools for testing a series (or the first or second difference of the series) for the presence of a unit root. In addition to the existing Augmented Dickey-Fuller (1979) and Phillips-Perron (1998) tests, EViews now allows you to compute the GLS-detrended Dickey-Fuller (Elliot, Rothenberg, and Stock, 1996), Kwiatkowski, Phillips, Schmidt, and Shin (KPSS, 1992), Elliott, Rothenberg, and Stock Point Optimal (ERS, 1996), and Ng and Perron (NP, 2001) unit root tests. All of these tests are available as a view of a series.

Performing Unit Root Tests in EViews

The following discussion assumes that you are familiar with the basic forms of the unit root tests and the associated options. We provide theoretical background for these tests in "Basic Unit Root Theory" beginning on page 521, and document the settings used when performing these tests.



To begin, double click on the series name to open the series window, and choose **View/Unit Root Test…**

You must specify four sets of options to carry out a unit root test. The first three settings (on the left-hand side of the dialog) determine the basic form of the unit root test. The fourth set of options (on the right-hand side of the dialog) consist of test-specific advanced settings. You only need concern yourself with these settings if you wish to customize the calculation of your unit root test.

First, you should use the topmost combo box to select the type of unit root test that you wish to perform. You may choose one of six tests: ADF, DFGLS, PP, KPSS, ERS, and NP.

Next, specify whether you wish to test for a unit root in the level, first difference, or second difference of the series.

Lastly, choose your exogenous regressors. You can choose to include a constant, a constant and linear trend, or neither (there are limitations on these choices for some of the tests).

You can click on **OK** to compute the test using the specified settings, or you can customize your test using the advanced settings portion of the dialog.

The advanced settings for both the ADF and DFGLS tests allow you to specify how lagged difference terms p are to be included in the ADF test equation. You may choose to let

EViews automatically select p, or you may specify a fixed positive integer value (if you choose automatic selection, you are given the additional option of selecting both the information criterion and maximum number of lags to be used in the selection procedure).

In this case, we have chosen to estimate an ADF test that includes a constant in the test regression and employs automatic lag length selection using a Schwarz Information Criterion (BIC) and a maximum lag length of 14. Applying these settings to data on the U. S. one-month Treasury bill rate for the period from March 1953 to July 1971, we can replicate Example 9.2 of Hayashi (2000, p. 596). The results are described below.

The first part of the unit root output provides information about the form of the test (the type of test, the exogenous variables, and lag length used), and contains the test output, associated critical values, and in this case, the p-value:

Null Hypothesis: TBILL has a unit root
Exogenous: Constant
Lag Length: 1 (Automatic based on SIC, MAXLAG=14)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-1.417410	0.5734
Test critical values:	1% level	-3.459898	
	5% level	-2.874435	
	10% level	-2.573719	

*MacKinnon (1996) one-sided p-values.

The ADF statistic value is -1.417 and the associated one-sided p-value (for a test with 221 observations) is .573. In addition, EViews reports the critical values at the 1%, 5% and 10% levels. Notice here that the statistic t_α value is greater than the critical values so that we do not reject the null at conventional test sizes.

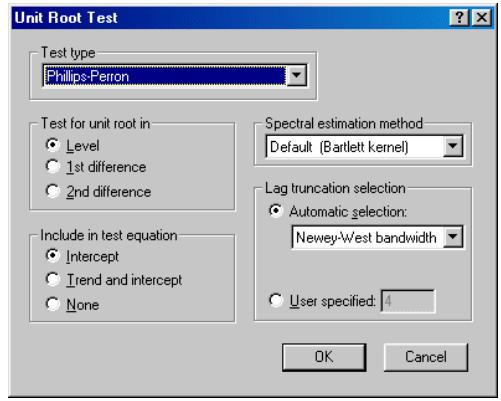
The second part of the output shows the intermediate test equation that EViews used to calculate the ADF statistic:

Augmented Dickey-Fuller Test Equation
Dependent Variable: D(TBILL)
Method: Least Squares
Date: 02/07/02 Time: 12:29
Sample: 1953:03 1971:07
Included observations: 221

Variable	Coefficient	Std. Error	t-Statistic	Prob.
TBILL(-1)	-0.022951	0.016192	-1.417410	0.1578
D(TBILL(-1))	-0.203330	0.067007	-3.034470	0.0027
C	0.088398	0.056934	1.552626	0.1220

R-squared	0.053856	Mean dependent var	0.013826
Adjusted R-squared	0.045175	S.D. dependent var	0.379758
S.E. of regression	0.371081	Akaike info criterion	0.868688
Sum squared resid	30.01882	Schwarz criterion	0.914817
Log likelihood	-92.99005	F-statistic	6.204410
Durbin-Watson stat	1.976361	Prob(F-statistic)	0.002395

If you had chosen to perform any of the other unit root tests (PP, KPSS, ERS, NP), the right side of the dialog would show the different options associated with the specified test. The options are associated with the method used to estimate the zero frequency spectrum term, f_0, that is used in constructing the particular test statistic. As before, you only need pay attention to these settings if you wish to change from the EViews defaults.



Here, we have selected the PP test in the combo box. Note that the right-hand side of the dialog has changed, and now features a combo box for selecting the spectral estimation method. You may use this combo box to choose between various kernel or AR regression based estimators for f_0. The entry labeled "Default" will show you the default estimator for the specific unit root test—in this example, we see that the PP default uses a kernel sum-of-covariances estimator with Bartlett weights. Alternately, if you had selected a NP test, the default entry would be "AR spectral-GLS".

Lastly, you can control the lag length or bandwidth used for your spectral estimator. If you select one of the kernel estimation methods (Bartlett, Parzen, Quadratic Spectral), the dialog will give you a choice between using Newey-West or Andrews automatic bandwidth selection methods, or providing a user specified bandwidth. If you choose one of the AR spectral density estimation methods (AR Spectral - OLS, AR Spectral - OLS detrended, AR Spectral - GLS detrended), the dialog will prompt you to choose from various automatic lag length selection methods (using information criteria) or to provide a user-specified lag length. See "Automatic Bandwidth and Lag Length Selection" on page 529.

Once you have chosen the appropriate settings for your test, click on the **OK** button. EViews reports the test statistic along with output from the corresponding test regression. For these tests, EViews reports the uncorrected estimate of the residual variance and the estimate of the frequency zero spectrum f_0 (labeled as the "HAC corrected variance") in addition to the basic output. Running a PP test using the TBILL series yields:

Null Hypothesis: TBILL has a unit root
Exogenous: Constant
Bandwidth: 3.82 (Andrews using Bartlett kernel)

		Adj. t-Stat	Prob.*
Phillips-Perron test statistic		-1.519035	0.5223
Test critical values:	1% level	-3.459898	
	5% level	-2.874435	
	10% level	-2.573719	

*MacKinnon (1996) one-sided p-values.

Residual variance (no correction)	0.141569
HAC corrected variance (Bartlett kernel)	0.107615

As with the ADF test, we fail to reject the null hypothesis of a unit root in the TBILL series at conventional significance levels.

Note that your test output will differ somewhat for alternative test specifications. For example, the KPSS output only provides the asymptotic critical values tabulated by KPSS:

Null Hypothesis: TBILL is stationary
Exogenous: Constant
Bandwidth: 11 (Newey-West Fixed using Bartlett kernel)

		LM-Stat.
Kwiatkowski-Phillips-Schmidt-Shin test statistic		1.537310
Asymptotic critical values*:	1% level	0.739000
	5% level	0.463000
	10% level	0.347000

*Kwiatkowski-Phillips-Schmidt-Shin (1992, Table 1)

Residual variance (no correction)	2.415060
HAC corrected variance (Bartlett kernel)	26.11028

Similarly, the NP test output will contain results for all four test statistics, along with the NP tabulated critical values.

A word of caution. You should note that the critical values reported by EViews are valid only for unit root tests of a data series, and will be invalid if the series is based on estimated values. For example, Engle and Granger (1987) proposed a two-step method to test for cointegration. The test amounts to testing for a unit root in the residuals of a first stage regression. Since these residuals are estimates of the disturbance term, the asymptotic distribution of the test statistic differs from the one for ordinary series. The correct critical values for a subset of the tests may be found in Davidson and MacKinnon (1993, Table 20.2).

Basic Unit Root Theory

The following discussion outlines the basics features of unit root tests. By necessity, the discussion will be brief. Users who require detail should consult the original sources and

standard references (see, for example, Davidson and MacKinnon, 1993, Chapter 20, Hamilton, 1994, Chapter 17, and Hayashi, 2000, Chapter 9).

Consider a simple AR(1) process:

$$y_t = \rho y_{t-1} + x_t'\delta + \epsilon_t, \tag{17.39}$$

where x_t are optional exogenous regressors which may consist of constant, or a constant and trend, ρ and δ are parameters to be estimated, and the ϵ_t are assumed to be white noise. If $|\rho| \geq 1$, y is a nonstationary series and the variance of y increases with time and approaches infinity. If $|\rho| < 1$, y is a (trend-)stationary series. Thus, the hypothesis of (trend-)stationarity can be evaluated by testing whether the absolute value of ρ is strictly less than one.

The unit root tests that EViews provides generally test the null hypothesis H_0: $\rho = 1$ against the one-sided alternative H_1: $\rho < 1$. In some cases, the null is tested against a point alternative. In contrast, the KPSS Lagrange Multiplier test evaluates the null of H_0: $\rho < 1$ against the alternative H_1: $\rho = 1$.

The Augmented Dickey-Fuller (ADF) Test

The standard DF test is carried out by estimating Equation (17.39) after subtracting y_{t-1} from both sides of the equation:

$$\Delta y_t = \alpha y_{t-1} + x_t'\delta + \epsilon_t, \tag{17.40}$$

where $\alpha = \rho - 1$. The null and alternative hypotheses may be written as,

$$\begin{aligned} H_0&: \alpha = 0 \\ H_1&: \alpha < 0 \end{aligned} \tag{17.41}$$

and evaluated using the conventional t-ratio for α:

$$t_\alpha = \hat{\alpha}/(se(\hat{\alpha})) \tag{17.42}$$

where $\hat{\alpha}$ is the estimate of α, and $se(\hat{\alpha})$ is the coefficient standard error.

Dickey and Fuller (1979) show that under the null hypothesis of a unit root, this statistic does not follow the conventional Student's t-distribution, and they derive asymptotic results and simulate critical values for various test and sample sizes. More recently, MacKinnon (1991, 1996) implements a much larger set of simulations than those tabulated by Dickey and Fuller. In addition, MacKinnon estimates response surfaces for the simulation results, permitting the calculation of Dickey-Fuller critical values and p-values for arbitrary sample sizes. The more recent MacKinnon critical value calculations are used by EViews in constructing test output.

The simple Dickey-Fuller unit root test described above is valid only if the series is an AR(1) process. If the series is correlated at higher order lags, the assumption of white noise

disturbances ϵ_t is violated. The Augmented Dickey-Fuller (ADF) test constructs a parametric correction for higher-order correlation by assuming that the y series follows an AR(p) process and adding p lagged difference terms of the dependent variable y to the right-hand side of the test regression:

$$\Delta y_t = \alpha y_{t-1} + x_t'\delta + \beta_1 \Delta y_{t-1} + \beta_2 \Delta y_{t-2} + \ldots + \beta_p \Delta y_{t-p} + v_t . \tag{17.43}$$

This augmented specification is then used to test (17.41) using the t-ratio (17.42). An important result obtained by Fuller is that the asymptotic distribution of the t-ratio for α is independent of the number of lagged first differences included in the ADF regression. Moreover, while the assumption that y follows an autoregressive (AR) process may seem restrictive, Said and Dickey (1984) demonstrate that the ADF test is asymptotically valid in the presence of a moving average (MA) component, provided that sufficient lagged difference terms are included in the test regression.

You will face two practical issues in performing an ADF test. First, you must choose whether to include exogenous variables in the test regression. You have the choice of including a constant, a constant and a linear time trend, or neither in the test regression. One approach would be to run the test with both a constant and a linear trend since the other two cases are just special cases of this more general specification. However, including irrelevant regressors in the regression will reduce the power of the test to reject the null of a unit root. The standard recommendation is to choose a specification that is a plausible description of the data under both the null and alternative hypotheses. See Hamilton (1994a, p. 501) for discussion.

Second, you will have to specify the number of lagged difference terms (which we will term the "lag length") to be added to the test regression (0 yields the standard DF test; integers greater than 0 correspond to ADF tests). The usual (though not particularly useful) advice is to include a number of lags sufficient to remove serial correlation in the residuals. EViews provides both automatic and manual lag length selection options. For details, see "Automatic Bandwidth and Lag Length Selection" beginning on page 529.

Dickey-Fuller Test with GLS Detrending (DFGLS)

As noted above, you may elect to include a constant, or a constant and a linear time trend, in your ADF test regression. For these two cases, ERS (1996) propose a simple modification of the ADF tests in which the data are detrended so that explanatory variables are "taken out" of the data prior to running the test regression.

ERS define a quasi-difference of y_t that depends on the value a representing the specific point alternative against which we wish to test the null:

$$d(y_t|a) = \begin{cases} y_t & \text{if } t = 1 \\ y_t - a y_{t-1} & \text{if } t > 1 \end{cases} \tag{17.44}$$

Next, consider an OLS regression of the quasi-differenced data $d(y_t|a)$ on the quasi-differenced $d(x_t|a)$:

$$d(y_t|a) = d(x_t|a)'\delta(a) + \eta_t \tag{17.45}$$

where x_t contains either a constant, or a constant and trend, and let $\hat{\delta}(a)$ be the OLS estimates from this regression.

All that we need now is a value for a. ERS recommend the use of $a = \bar{a}$, where:

$$\bar{a} = \begin{cases} 1 - 7/T & \text{if } x_t = \{1\} \\ 1 - 13.5/T & \text{if } x_t = \{1, t\} \end{cases} \tag{17.46}$$

We now define the *GLS detrended data,* y_t^d using the estimates associated with the $\bar{a}$:

$$y_t^d \equiv y_t - x_t'\hat{\delta}(\bar{a}) \tag{17.47}$$

Then the DFGLS test involves estimating the standard ADF test equation, (17.43), after substituting the GLS detrended y_t^d for the original y_t:

$$\Delta y_t^d = \alpha y_{t-1}^d + \beta_1 \Delta y_{t-1}^d + \ldots + \beta_p y_{t-p}^d + v_t \tag{17.48}$$

Note that since the y_t^d are detrended, we do not include the x_t in the DFGLS test equation. As with the ADF test, we consider the t-ratio for $\hat{\alpha}$ from this test equation.

While the DFGLS t-ratio follows a Dickey-Fuller distribution in the constant only case, the asymptotic distribution differs when you include both a constant and trend. ERS (1996, Table 1, p. 825) simulate the critical values of the test statistic in this latter setting for $T = \{50, 100, 200, \infty\}$. Thus, the EViews lower tail critical values use the MacKinnon simulations for the no constant case, but are interpolated from the ERS simulated values for the constant and trend case. The null hypothesis is rejected for values that fall below these critical values.

The Phillips-Perron (PP) Test

Phillips and Perron (1988) propose an alternative (nonparametric) method of controlling for serial correlation when testing for a unit root. The PP method estimates the non-augmented DF test equation (17.40), and modifies the t-ratio of the α coefficient so that serial correlation does not affect the asymptotic distribution of the test statistic. The PP test is based on the statistic:

$$\tilde{t}_\alpha = t_\alpha \left(\frac{\gamma_0}{f_0}\right)^{1/2} - \frac{T(f_0 - \gamma_0)(se(\hat{\alpha}))}{2 f_0^{1/2} s} \tag{17.49}$$

where $\hat{\alpha}$ is the estimate, and t_α the t-ratio of α, $se(\hat{\alpha})$ is coefficient standard error, and s is the standard error of the test regression. In addition, γ_0 is a consistent estimate of the

error variance in (17.40) (calculated as $(T-k)s^2/T$, where k is the number of regressors). The remaining term, f_0, is an estimator of the residual spectrum at frequency zero.

There are two choices you will have make when performing the PP test. First, you must choose whether to include a constant, a constant and a linear time trend, or neither, in the test regression. Second, you will have to choose a method for estimating f_0. EViews supports estimators for f_0 based on kernel-based sum-of-covariances, or on autoregressive spectral density estimation. See "Frequency Zero Spectrum Estimation" beginning on page 527 for details.

The asymptotic distribution of the PP modified t-ratio is the same as that of the ADF statistic. EViews reports MacKinnon lower-tail critical and p-values for this test.

The Kwiatkowski, Phillips, Schmidt, and Shin (KPSS) Test

The KPSS (1992) test differs from the other unit root tests described here in that the series y_t is assumed to be (trend-) stationary under the null. The KPSS statistic is based on the the residuals from the OLS regression of y_t on the exogenous variables x_t:

$$y_t = x_t'\delta + u_t \tag{17.50}$$

The LM statistic is be defined as:

$$LM = \sum_t S(t)^2/(T^2 f_0) \tag{17.51}$$

where f_0, is an estimator of the residual spectrum at frequency zero and where $S(t)$ is a cumulative residual function:

$$S(t) = \sum_{r=1}^{t} \hat{u}_r \tag{17.52}$$

based on the residuals $\hat{u}_t = y_t - x_t'\hat{\delta}(0)$. We point out that the estimator of δ used in this calculation differs from the estimator for δ used by GLS detrending since it is based on a regression involving the original data and not on the quasi-differenced data.

To specify the KPSS test, you must specify the set of exogenous regressors x_t and a method for estimating f_0. See "Frequency Zero Spectrum Estimation" on page 527 for discussion.

The reported critical values for the LM test statistic are based upon the asymptotic results presented in KPSS (Table 1, p. 166).

Elliot, Rothenberg, and Stock Point Optimal (ERS) Test

The ERS Point Optimal test is based on the quasi-differencing regression defined in Equations (17.45). Define the residuals from (17.45) as $\hat{\eta}_t(a) = d(y_t|a) - d(x_t|a)'\hat{\delta}(a)$, and let $\text{SSR}(a) = \sum\hat{\eta}_t^2(a)$ be the sum-of-squared residuals function. The ERS (feasible)

point optimal test statistic of the null that $\alpha = 1$ against the alternative that $\alpha = \bar{a}$, is then defined as:

$$P_T = (\text{SSR}(\bar{a}) - \bar{a}\text{SSR}(1))/f_0 \tag{17.53}$$

where f_0, is an estimator of the residual spectrum at frequency zero.

To compute the ERS test, you must specify the set of exogenous regressors x_t and a method for estimating f_0 (see "Frequency Zero Spectrum Estimation" on page 527).

Critical values for the ERS test statistic are computed by interpolating the simulation results provided by ERS (1996, Table 1, p. 825) for $T = \{50, 100, 200, \infty\}$.

Ng and Perron (NP) Tests

Ng and Perron (2001) construct four test statistics that are based upon the GLS detrended data y_t^d. These test statistics are modified forms of Phillips and Perron Z_α and Z_t statistics, the Bhargava (1986) R_1 statistic, and the ERS Point Optimal statistic. First, define the term:

$$\kappa = \sum_{t=2}^{T} (y_{t-1}^d)^2 / T^2 \tag{17.54}$$

The modified statistics may then be written as,

$$\begin{aligned}
MZ_\alpha^d &= (T^{-1}(y_T^d)^2 - f_0)/(2\kappa) \\
MZ_t^d &= MZ_\alpha \times MSB \\
MSB^d &= (\kappa/f_0)^{1/2} \\
MP_T^d &= \begin{cases} (\bar{c}^2\kappa - \bar{c}T^{-1}(y_T^d)^2)/f_0 & \text{if } x_t = \{1\} \\ (\bar{c}^2\kappa + (1-\bar{c})T^{-1}(y_T^d)^2)/f_0 & \text{if } x_t = \{1, t\} \end{cases}
\end{aligned} \tag{17.55}$$

where:

$$\bar{c} = \begin{cases} -7 & \text{if } x_t = \{1\} \\ -13.5 & \text{if } x_t = \{1, t\} \end{cases} \tag{17.56}$$

The NP tests require a specification for x_t and a choice of method for estimating f_0 (see "Frequency Zero Spectrum Estimation" on page 527).

Frequency Zero Spectrum Estimation

Many of the unit root tests described above require a consistent estimate of the residual spectrum at frequency zero. EViews supports two classes of estimators for f_0: kernel-based sum-of-covariances estimators, and autoregressive spectral density estimators.

Kernel Sum-of-Covariances Estimation

The kernel-based estimator of the frequency zero spectrum is based on a weighted sum of the autocovariances, with the weights are defined by a kernel function. The estimator takes the form,

$$\hat{f}_0 = \sum_{j=-(T-1)}^{T-1} \hat{\gamma}(j) \cdot K(j/l) \tag{17.57}$$

where l is a bandwidth parameter (which acts as a truncation lag in the covariance weighting), K is a kernel function, and where $\hat{\gamma}(j)$, the j-th sample autocovariance of the residuals $\tilde{u}_t$, is defined as:

$$\hat{\gamma}(j) = \sum_{t=j+1}^{T} (\tilde{u}_t \tilde{u}_{t-j})/T \tag{17.58}$$

Note that the residuals $\tilde{u}_t$ that EViews uses in estimating the autocovariance functions in (17.58) will differ depending on the specified unit root test:

Unit root test	**Source of $\tilde{u}_t$ residuals for kernel estimator**
ADF, DFGLS	*not applicable.*
PP, ERS Point Optimal, NP	residuals from the Dickey-Fuller test equation, (17.40).
KPSS	residuals from the OLS test equation, (17.50).

EViews supports the following kernel functions:

Bartlett:	$K(x) = \begin{cases} 1-\lvert x\rvert & \text{if } \lvert x\rvert \le 1 \\ 0 & \text{otherwise} \end{cases}$

Parzen:	$K(x) = \begin{cases} 1 - 6x^2 + 6\lvert x\rvert^3 & \text{if } 0 \le \lvert x\rvert \le (1/2) \\ 2(1 - \lvert x\rvert)^3 & \text{if } (1/2) < \lvert x\rvert \le 1 \\ 0 & \text{otherwise} \end{cases}$
Quadratic Spectral	$K(x) = \frac{25}{12\pi^2 x^2}\left(\frac{\sin(6\pi x/5)}{6\pi x/5} - \cos(6\pi x/5)\right)$

The properties of these kernels are described in Andrews (1991).

As with most kernel estimators, the choice of the bandwidth parameter l is of considerable importance. EViews allows you to specify a fixed parameter or to have EViews select one using a data-dependent method. Automatic bandwidth parameter selection is discussed in "Automatic Bandwidth and Lag Length Selection" beginning on page 529.

Autoregressive Spectral Density Estimator

The autoregressive spectral density estimator at frequency zero is based upon the residual variance and estimated coefficients from the auxiliary regression:

$$\Delta\tilde{y}_t = \alpha\tilde{y}_{t-1} + \varphi \cdot \tilde{x}_t'\delta + \beta_1\Delta\tilde{y}_{t-1} + \ldots + \beta_p\Delta\tilde{y}_{t-p} + u_t \tag{17.59}$$

EViews provides three autoregressive spectral methods: OLS, OLS detrending, and GLS detrending, corresponding to difference choices for the data $\tilde{y}_t$. The following table summarizes the auxiliary equation estimated by the various AR spectral density estimators:

AR spectral method	**Auxiliary AR regression specification**
OLS	$\tilde{y}_t = y_t$, and $\varphi = 1$, $\tilde{x}_t = x_t$.
OLS detrended	$\tilde{y}_t = y_t - x_t'\hat{\delta}(0)$, and $\varphi = 0$.
GLS detrended	$\tilde{y}_t = y_t - x_t'\hat{\delta}(\bar{a}) = y_t^d$. and $\varphi = 0$.

where $\hat{\delta}(a)$ are the coefficient estimates from the regression defined in (17.45).

The AR spectral estimator of the frequency zero spectrum is defined as:

$$\hat{f}_0 = \hat{\sigma}_u^2/(1 - \hat{\beta}_1 - \hat{\beta}_2 - \ldots - \hat{\beta}_p) \tag{17.60}$$

where $\hat{\sigma}_u^2 = \sum\tilde{u}_t^2/T$ is the residual variance, and $\hat{\beta}$ are the estimates from (17.59). We note here that EViews uses the non-degree of freedom estimator of the residual variance. As a result, spectral estimates computed in EViews may differ slightly from those obtained from other sources.

Not surprisingly, the spectrum estimator is sensitive to the number of lagged difference terms in the auxiliary equation. You may either specify a fixed parameter or have EViews

automatically select one based on an information criterion. Automatic lag length selection is examined in "Automatic Bandwidth and Lag Length Selection" on page 529.

Default Settings

By default, EViews will choose the estimator of f_0 used by the authors of a given test specification. You may, of course, override the default settings and choose from either family of estimation methods. The default settings are listed below:

Unit root test	**Frequency zero spectrum default method**
ADF, DFGLS	*not applicable*
PP, KPSS	Kernel (Bartlett) sum-of-covariances
ERS Point Optimal	AR spectral regression (OLS)
NP	AR spectral regression (GLS-detrended)

Automatic Bandwidth and Lag Length Selection

There are three distinct situations in which EViews can automatically compute a bandwidth or a lag length parameter.

The first situation occurs when you are selecting the bandwidth parameter l for the kernel-based estimators of f_0. For the kernel estimators, EViews provides you with the option of using the Newey-West (1994) or the Andrews (1991) data-based automatic bandwidth parameter methods. See the original sources for details. For those familiar with the Newey-West procedure, we note that EViews uses the lag selection parameter formulae given in the corresponding first lines of Table II-C. The Andrews method is based on an AR(1) specification.

The latter two occur when the unit root test requires estimation of a regression with a parametric correction for serial correlation as in the ADF and DFGLS test equation regressions, and in the AR spectral estimator for f_0. In all of these cases, p lagged difference terms are added to a regression equation. The automatic selection methods choose p (less than the specified maximum) to minimize one of the following criteria:

Information criterion	**Definition**
Akaike (AIC)	$-2(l/T) + 2k/T$
Schwarz (SIC)	$-2(l/T) + k\log(T)/T$
Hannan-Quinn (HQ)	$-2(l/T) + 2k\log(\log(T))/T$
Modified AIC (MAIC)	$-2(l/T) + 2(k+\tau)/T$
Modified SIC (MSIC)	$-2(l/T) + (k+\tau)\log(T)/T$

Modified Hannan-Quinn (MHQ)	$-2(l/T) + 2(k+\tau)\log(\log(T))/T$

where the modification factor τ is computed as:

$$\tau = \alpha^2 \sum_t \tilde{y}_{t-1}^2 / \hat{\sigma}_u^2 \tag{17.61}$$

for $\tilde{y}_t = y_t$, when computing the ADF test equation, and for $\tilde{y}_t$ as defined in "Autoregressive Spectral Density Estimator" on page 528, when estimating f_0. NP (2001) propose and examine the modified criteria, concluding with a recommendation of the MAIC.

For the information criterion selection methods, you must also specify an upper bound to the lag length. By default, EViews chooses a maximum lag of:

$$k_{\max} = \text{int}(12(T/100)^{1/4}) \tag{17.62}$$

See Hayashi (2000, p. 594) for a discussion of the selection of this upper bound.

Panel Unit Root Tests

Recent literature suggests that panel-based unit root tests have higher power than unit root tests based on individual time series. EViews will compute one of the following five types of panel unit root tests: Levin, Lin and Chu (2002), Breitung (2000), Im, Pesaran and Shin (2003), Fisher-type tests using ADF and PP tests (Maddala and Wu (1999) and Choi (2001)), and Hadri (1999).

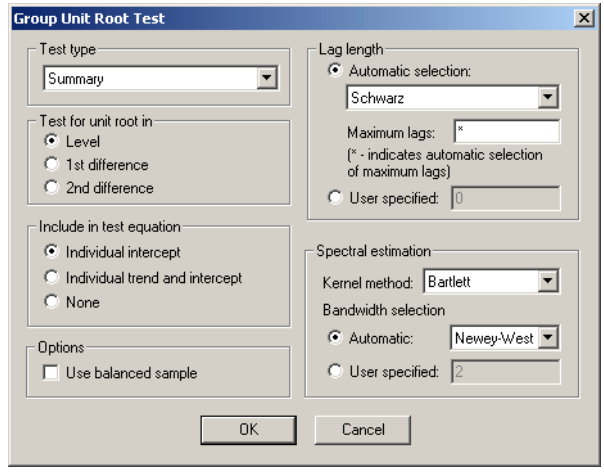
While these tests are commonly termed "panel unit root" tests, theoretically, they are simply multiple-series unit root tests that have been applied to panel data structures (where the presence of cross-sections generates "multiple series" out of a single series). Accordingly, EViews supports these tests in settings involving multiple series: as a series view (if the workfile is panel structured), as a group view, or as a pool view.

Performing Panel Unit Root Tests in EViews

The following discussion assumes that you are familiar with the basics of both unit root tests and panel unit root tests.

To begin, select **View/Unit Root Test...**from the menu of an EViews group or pool object, or from the menu of an individual series in a panel structured workfile. Here we show the dialog for a Group unit root test—the other dialogs differ slightly (for the pool object, there is an additional field in the upper-left hand portion of the dialog where you must indicate the name of the pool series on which you wish to conduct your test; for the series object in a panel workfile, the "balanced sample" option is not present).

If you wish to accept the default settings, simply click on **OK.** EViews will use the default **Summary** setting, and will compute a full suite of unit root tests on the levels of the series, along with a summary of the results.



To customize the unit root calculations, you will choose from a variety of options. The options on the left-hand side of the dialog determine the basic structure of the test or tests, while the options on the right-hand side of the dialog control advanced computational details such as bandwidth or lag selection methods, or kernel methods.

The combo box at the top of the dialog is where you will choose the type of test to perform. There are six settings: "**Summary",** "**Common root - Levin, Lin, Chu**", "**Common root - Breitung**", "**Individual root - Im, Pesaran, Shin"**, "**Individual root - Fisher - ADF"**, "**Individual root - Fisher - PP"**, and "**Common root - Hadri"**, corresponding to one or more of the tests listed above. The combo box labels include a brief description of the assumptions under which the tests are computed. "Common root" indicates that the tests are estimated assuming a common AR structure for all of the series; "Individual root" is used for tests which allow for different AR coefficients in each series.

We have already pointed out that the **Summary** default instructs EViews to estimate all of the tests, and to provide a brief summary of the results. Selecting an individual test type allows you better control over the computational method and provides additional detail on the test results.

The next two sets of radio buttons allow you to control the specification of your test equation. First, you may choose to conduct the unit root on the **Level**, **1st difference**, or **2nd difference** of your series. Next, you may choose between sets of exogenous regressors to be included. You can select **Individual intercept** if you wish to include individual fixed effects, **Individual intercepts and individual trends** to include both fixed effects and trends, or **None** for no regressors.

The **Use balanced sample** option is present only if you are estimating a Pool or a Group unit root test. If you select this option, EViews will adjust your sample so that only observations where all series values are not missing will be included in the test equations.

Depending on the form of the test or tests to be computed, you will be presented with various advanced options on the right side of the dialog. For tests that involve regressions on lagged difference terms (Levin, Lin, and Chu, Breitung, Im, Pesaran, and Shin, Fisher - ADF) these options relate to the choice of the number of lags to be included. For the tests involving kernel weighting (Levin, Lin, and Chu, Fisher - PP, Hadri), the options relate to the choice of bandwidth and kernel type.

For a group or pool unit root test, the EViews default is to use automatic selection methods: information matrix criterion based for the number of lag difference terms (with automatic selection of the maximum lag to evaluate), and the Andrews or Newey-West method for bandwidth selection. For unit root tests on a series in a panel workfile, the default behavior uses user-specified options.

If you wish to override these settings, simply enter the appropriate information. You may, for example, select a fixed, user-specified number of lags by entering a number in the **User specified** field. Alternatively, you may customize the settings for automatic lag selection method. Alternative criteria for evaluating the optimal lag length may be selected via the combo box (Akaike, Schwarz, Hannan-Quinn, Modified Akaike, Modified Schwarz, Modified Hannan-Quinn), and you may limit the number of lags to try in automatic selection by entering a number in the **Maximum lags** box. For the kernel based methods, you may select a kernel type from the combo box (**Bartlett**, **Parzen**, **Quadratic spectral**), and you may specify either an automatic bandwidth selection method (**Andrews**, **Newey-West**) or user-specified fixed bandwidth.

As an illustration, we compute the summary panel unit root test, using individual fixed effects as regressors, and automatic lag difference term and bandwidth selection (using the Schwarz criterion for the lag differences, and the Newey-West method and the Bartlett kernel for the bandwidth). The results for the panel unit root test are presented below:

Panel unit root test: Summary
Date: 07/02/03 Time: 13:13
Sample: 1935 1954
Exogenous variables: Individual effects
Automatic selection of maximum lags
Automatic selection of lags based on SIC: 0 to 3
Newey-West bandwidth selection using Bartlett kernel

Method	Statistic	Prob.**	Cross-sections	Obs
Null: Unit root (assumes common unit root process)				
Levin, Lin & Chu t*	2.39544	0.9917	10	184
Breitung t-stat	-2.06574	0.0194	10	174
Null: Unit root (assumes individual unit root process)				
Im, Pesaran and Shin W-stat	2.80541	0.9975	10	184
ADF - Fisher Chi-square	12.0000	0.9161	10	184
PP - Fisher Chi-square	12.9243	0.8806	10	190
Null: No unit root (assumes common unit root process)				
Hadri Z-stat	9.33436	0.0000	10	200

** Probabilities for Fisher tests are computed using an asymptotic Chi -square distribution. All other tests assume asymptotic normality.

The top of the output indicates the type of test, exogenous variables and test equation options. If we were instead estimating a Pool or Group test, a list of the series used in the test would also be depicted. The lower part of the summary output gives the main test results, organized both by null hypothesis as well as the maintained hypothesis concerning the type of unit root process. For example, we group the results for the LLC and the Breitung tests, since they both have a null of a unit root for the common process.

For the most part, the results indicate the presence of a unit root. The LLC, IPS, and both Fisher tests fail to reject the null of a unit root. Similarly, the Hadri test statistic, which tests the null of no unit root, strongly rejects the null in favor of a unit root. The one exception to this pattern is the Breitung test, which does reject the unit root null.

If you only wish to compute a single unit root test type, or if you wish to examine the tests results in greater detail, you may simply repeat the unit root test after selecting the desired test in **Test type** combo box. Here, we show the bottom portion of the LLC test specific output for the same data:

Intermediate results on I

Cross section	2nd Stage Coefficient	Variance of Reg	HAC of Dep.	Lag	Max Lag	Band-width	Obs
1	-0.35898	147.58	11.767	1	4	18.0	18
2	-0.05375	444.60	236.40	0	4	7.0	19
3	-0.11741	0.8153	0.5243	0	4	5.0	19
4	-0.10233	408.12	179.68	3	4	5.0	16
5	0.22672	11314.	18734.	0	4	1.0	19
6	-0.26332	90.040	89.960	0	4	2.0	19
7	0.12362	62.429	82.716	0	4	1.0	19
8	-0.13862	129.04	22.173	0	4	17.0	19
9	-0.55912	7838.8	1851.4	1	4	11.0	18
10	-0.44416	113.56	43.504	1	4	6.0	18

	Coefficient	t-Stat	SE Reg	mu*	sig*		Obs
Pooled	-0.01940	-0.464	1.079	-0.554	0.919		184

For each cross-section, the autoregression coefficient, variance of the regression, HAC of the dependent variable, the selected lag order, maximum lag, bandwidth truncation parameter, and the number of observations used are displayed.

Panel Unit Root Details

Panel unit root tests are similar, but not identical, to unit root tests carried out on a single series. Here, we briefly describe the five panel unit root tests currently supported in EViews; for additional detail, we encourage you to consult the original literature. The discussion assumes that you have a basic knowledge of unit root theory.

We begin by classifying our unit root tests on the basis of whether there are restrictions on the autoregressive process across cross-sections or series. Consider a following AR(1) process for panel data:

$$y_{it} = \rho_i y_{it-1} + X_{it}\delta_i + \epsilon_{it} \tag{17.63}$$

where $i = 1, 2, \ldots, N$ cross-section units or series, that are observed over periods $t = 1, 2, \ldots, T_i$.

The X_{it} represent the exogenous variables in the model, including any fixed effects or individual trends, ρ_i are the autoregressive coefficients, and the errors ϵ_{it} are assumed to be mutually independent idiosyncratic disturbance. If $|\rho_i| < 1$, y_i is said to be weakly (trend-) stationary. On the other hand, if $|\rho_i| = 1$ then y_i contains a unit root.

For purposes of testing, there are two natural assumptions that we can make about the ρ_i. First, one can assume that the persistence parameters are common across cross-sections so

that $\rho_i = \rho$ for all i. The Levin, Lin, and Chu (LLC), Breitung, and Hadri tests all employ this assumption. Alternatively, one can allow ρ_i to vary freely across cross-sections. The Im, Pesaran, and Shin (IPS), and Fisher-ADF and Fisher-PP tests are of this form.

Tests with Common Unit Root Process

Levin, Lin, and Chu (LLC), Breitung, and Hadri tests all assume that there is a common unit root process so that ρ_i is identical across cross-sections. The first two tests employ a null hypothesis of a unit root while the Hadri test uses a null of no unit root.

LLC and Breitung both consider the following basic ADF specification:

$$\Delta y_{it} = \alpha y_{it-1} + \sum_{j=1}^{p_i} \beta_{ij} \Delta y_{it-j} + X'_{it}\delta + \epsilon_{it} \tag{17.64}$$

where we assume a common $\alpha = \rho - 1$, but allow the lag order for the difference terms, p_i, to vary across cross-sections. The null and alternative hypotheses for the tests may be written as:

$$H_0\colon \alpha = 0 \tag{17.65}$$

$$H_1\colon \alpha < 0 \tag{17.66}$$

Under the null hypothesis, there is a unit root, while under the alternative, there is no unit root.

Levin, Lin, and Chu

The method described in LLC derives estimates of α from proxies for Δy_{it} and y_{it} that are standardized and free of autocorrelations and deterministic components.

For a given set of lag orders, we begin by estimating two additional sets of equations, regressing both Δy_{it}, and y_{it-1} on the lag terms Δy_{it-j} (for $j = 1, \ldots, p_i$) and the exogenous variables X_{it}. The estimated coefficients from these two regressions will be denoted $(\hat{\beta}, \hat{\delta})$ and $(\dot{\beta}, \dot{\delta})$, respectively.

We define $\Delta \bar{y}_{it}$ by taking Δy_{it} and removing the autocorrelations and deterministic components using the first set of auxiliary estimates:

$$\Delta \bar{y}_{it} = \Delta y_{it} - \sum_{j=1}^{p_i} \hat{\beta}_{ij} \Delta y_{it-j} - X'_{it}\hat{\delta} \tag{17.67}$$

Likewise, we may define the analogous $\bar{y}_{it-1}$ using the second set of coefficients:

$$\bar{y}_{it-1} = y_{it-1} - \sum_{j=1}^{p_i} \dot{\beta}_{ij} \Delta y_{it-j} - X'_{it}\dot{\delta} \tag{17.68}$$

Next, we obtain our proxies by standardizing both $\Delta\bar{y}_{it}$ and $\bar{y}_{it-1}$, dividing by the regression standard error:

$$\begin{aligned}\Delta\tilde{y}_{it} &= (\Delta\bar{y}_{it}/s_i)\\ \tilde{y}_{it-1} &= (\bar{y}_{it-1}/s_i)\end{aligned} \tag{17.69}$$

where s_i are the estimated standard errors from estimating each ADF in Equation (17.64).

Lastly, an estimate of the coefficient α may be obtained from the pooled proxy equation:

$$\Delta\tilde{y}_{it} = \alpha\tilde{y}_{it-1} + \eta_{it} \tag{17.70}$$

LLC show that under the null, a modified t-statistic for the resulting $\hat{\alpha}$ is asymptotically normally distributed

$$t_\alpha^* = \frac{t_\alpha - (N\tilde{T})S_N\hat{\sigma}^{-2}se(\hat{\alpha})\mu_{m\tilde{T}}^*}{\sigma_{m\tilde{T}}^*} \rightarrow N(0,1) \tag{17.71}$$

where t_α is the standard t-statistic for $\hat{\alpha} = 0$, $\hat{\sigma}^2$ is the estimated variance of the error term η, $se(\hat{\alpha})$ is the standard error of $\hat{\alpha}$, and:

$$\tilde{T} = T - (\sum_i p_i/N) - 1 \tag{17.72}$$

The remaining terms, which involve complicated moment calculations, are described in greater detail in LLC . The average standard deviation ratio, S_N, is defined as the mean of the ratios of the long-run standard deviation to the innovation standard deviation for each individual. Its estimate is derived using kernel-based techniques. The remaining two terms, $\mu_{m\tilde{T}}^*$ and $\sigma_{m\tilde{T}}^*$ are adjustment terms for the mean and standard deviation.

The LLC method requires a specification of the number of lags used in each cross-section ADF regression, p_i, as well as kernel choices used in the computation of S_N. In addition, you must specify the exogenous variables used in the test equations. You may elect to include no exogenous regressors, or to include individual constant terms (fixed effects), or to employ individual constants and trends.

Breitung

The Breitung method differs from LLC in two distinct ways. First, only the autoregressive portion (and not the exogenous components) is removed when constructing the standardized proxies:

$$\begin{aligned}\Delta\tilde{y}_{it} &= \left(\Delta y_{it} - \sum_{j=1}^{p_i}\hat{\beta}_{ij}\Delta y_{it-j}\right)/s_i\\ \tilde{y}_{it-1} &= \left(y_{it-1} - \sum_{j=1}^{p_i}\hat{\beta}_{ij}\Delta y_{it-j}\right)/s_i\end{aligned} \tag{17.73}$$

where $\hat{\beta}$, $\dot{\beta}$, and s_i are as defined for LLC.

Second, the proxies are transformed and detrended,

$$\begin{aligned} \Delta y_{it}^* &= \sqrt{\frac{(T-t)}{(T-t+1)}}\left(\Delta\tilde{y}_{it} - \frac{\Delta\tilde{y}_{it+1} + \ldots + \Delta\tilde{y}_{it+T}}{T-t}\right) \\ y_{it-1}^* &= \tilde{y}_{it-1} - c_{it} \end{aligned} \tag{17.74}$$

where,

$$c_{it} = \begin{cases} 0 & \text{if no intercept or trend} \\ \tilde{y}_{i1} & \text{with intercept, no trend} \\ \tilde{y}_{i1} - ((t-1)/T)\tilde{y}_{iT} & \text{with intercept and trend} \end{cases} \tag{17.75}$$

The persistence parameter α is estimated from the pooled proxy equation:

$$\Delta y_{it}^* = \alpha y_{it-1}^* + \nu_{it} \tag{17.76}$$

Breitung shows that under the null, the resulting estimator α^* is asymptotically distributed as a standard normal.

The Breitung method requires only a specification of the number of lags used in each cross-section ADF regression, p_i, and the exogenous regressors. As with the LLC test, you may elect to include no exogenous regressors, or to include individual constant terms (fixed effects), or individual constants and trends. Note that in contrast with LLC, no kernel computations are required.

Hadri

The Hadri panel unit root test is similar to the KPSS unit root test, and has a null hypothesis of no unit root in any of the series in the panel. Like the KPSS test, the Hadri test is based on the residuals from the individual OLS regressions of y_{it} on a constant, or on a constant and a trend. For example, if we include both the constant and a trend, we derive estimates from:

$$y_{it} = \delta_i + \eta_i t + \epsilon_{it} \tag{17.77}$$

Given the residuals $\hat{\epsilon}$ from the individual regressions, we form the LM statistic:

$$LM_1 = \frac{1}{N}\left(\sum_{i=1}^{N}\left(\sum_t S_i(t)^2/T^2\right)/\bar{f}_0\right) \tag{17.78}$$

where $S_i(t)$ are the cumulative sums of the residuals,

$$S_i(t) = \sum_{s=1}^{t} \hat{\epsilon}_{it} \tag{17.79}$$

and $\bar{f}_0$ is the average of the individual estimators of the residual spectrum at frequency zero:

$$\bar{f}_0 = \sum_{i=1}^{N} f_{i0}/N \tag{17.80}$$

EViews provides several methods for estimating the f_{i0}. See "Unit Root Tests" on page 518 for additional details.

An alternative form of the LM statistic allows for heteroskedasticity across i:

$$LM_2 = \frac{1}{N}\left(\sum_{i=1}^{N}\left(\sum_t S_i(t)^2/T^2\right)/f_{i0}\right) \tag{17.81}$$

Hadri shows that under mild assumptions,

$$Z = \frac{\sqrt{N}(LM - \xi)}{\zeta} \rightarrow N(0,1) \tag{17.82}$$

where $\xi = 1/6$ and $\zeta = 1/45$, if the model only includes constants (η_i is set to 0 for all i), and $\xi = 1/15$ and $\zeta = 11/6300$, otherwise.

The Hadri panel unit root tests require only the specification of the form of the OLS regressions: whether to include only individual specific constant terms, or whether to include both constant and trend terms. EViews reports two Z-statistic values, one based on LM_1 with the associated homoskedasticity assumption, and the other using LM_2 that is heteroskedasticity consistent.

Tests with Individual Unit Root Processes

The Im, Pesaran, and Shin, and the Fisher-ADF and PP tests all allow for individual unit root processes so that ρ_i may vary across cross-sections. The tests are all characterized by the combining of individual unit root tests to derive a panel-specific result.

Im, Pesaran, and Shin

Im, Pesaran, and Shin begin by specifying a separate ADF regression for each cross section:

$$\Delta y_{it} = \alpha y_{it-1} + \sum_{j=1}^{p_i} \beta_{ij}\Delta y_{it-j} + X'_{it}\delta + \epsilon_{it} \tag{17.83}$$

The null hypothesis may be written as,

$$H_0\colon\ \alpha_i = 0, \text{ for all } i \tag{17.84}$$

while the alternative hypothesis is given by:

$$H_1: \begin{cases} \alpha_i = 0 & \text{for } i = 1, 2, \ldots, N_1 \\ \alpha_i < 0 & \text{for } i = N+1, N+2, \ldots, N \end{cases} \tag{17.85}$$

(where the i may be reordered as necessary) which may be interpreted as a non-zero fraction of the individual processes is stationary.

After estimating the separate ADF regressions, the average of the t-statistics for α_i from the individual ADF regressions, $t_{iT_i}(p_i)$:

$$\bar{t}_{NT} = \left(\sum_{i=1}^{N} t_{iT_i}(p_i) \right) / N \tag{17.86}$$

is then adjusted to arrive at the desired test statistics.

In the case where the lag order is always zero ($p_i = 0$ for all i), simulated critical values for $\bar{t}_{NT}$ are provided in the IPS paper for different numbers of cross sections N, series lengths T, and for test equations containing either intercepts, or intercepts and linear trends. EViews uses these values, or linearly interpolated values, in evaluating the significance of the test statistics.

In the general case where the lag order in Equation (17.83) may be non-zero for some cross-sections, IPS show that a properly standardized $\bar{t}_{NT}$ has an asymptotic standard normal distribution:

$$W_{\bar{t}_{NT}} = \frac{\sqrt{N}\left(\bar{t}_{NT} - N^{-1} \sum_{i=1}^{N} E(\bar{t}_{iT}(p_i)) \right)}{\sqrt{N^{-1} \sum_{i=1}^{N} Var(\bar{t}_{iT}(p_i))}} \rightarrow N(0,1) \tag{17.87}$$

The expressions for the expected mean and variance of the ADF regression t-statistics, $E(\bar{t}_{iT}(p_i))$ and $Var(\bar{t}_{iT}(p_i))$, are provided by IPS for various values of T and p and differing test equation assumptions, and are not provided here.

The IPS test statistic requires specification of the number of lags and the specification of the deterministic component for each cross-section ADF equation. You may choose to include individual constants, or to include individual constant and trend terms.

Fisher-ADF and Fisher-PP

An alternative approach to panel unit root tests uses Fisher's (1932) results to derive tests that combine the p-values from individual unit root tests. This idea has been proposed by Maddala and Wu, and by Choi.

If we define π_i as the p-value from any individual unit root test for cross-section i, then under the null of unit root for all N cross-sections, we have the asymptotic result that

$$-2 \sum_{i=1}^{N} \log(\pi_i) \rightarrow \chi^2_{2N} \tag{17.88}$$

In addition, Choi demonstrates that:

$$Z = \frac{1}{\sqrt{N}} \sum_{i=1}^{N} \Phi^{-1}(\pi_i) \rightarrow N(0, 1) \tag{17.89}$$

where Φ^{-1} is the inverse of the standard normal cumulative distribution function.

EViews reports both the asymptotic χ^2 and standard normal statistics using ADF and Phillips-Perron individual unit root tests. The null and alternative hypotheses are the same as for the as IPS.

For both Fisher tests, you must specify the exogenous variables for the test equations. You may elect to include no exogenous regressors, to include individual constants (effects), or include individual constant and trend terms.

Additionally, when the Fisher tests are based on ADF test statistics, you must specify the number of lags used in each cross-section ADF regression. For the PP form of the test, you must instead specify a method for estimating f_0. EViews supports estimators for f_0 based on kernel-based sum-of-covariances. See "Frequency Zero Spectrum Estimation" beginning on page 527 for details.

Summary of Available Panel Unit Root Tests

The following table summarizes the basic characteristics of the panel unit root tests available in EViews:

Test	Null	Alternative	Possible Deterministic Component	Autocorrelation Correction Method
Levin, Lin and Chu	Unit root	No Unit Root	None, F, T	Lags
Breitung	Unit root	No Unit Root	None, F, T	Lags
IPS	Unit Root	Some cross-sections without UR	F, T	Lags

Fisher-ADF	Unit Root	Some cross-sections without UR	None, F, T	Lags
Fisher-PP	Unit Root	Some cross-sections without UR	None, F, T	Kernel
Hadri	No Unit Root	Unit Root	F, T	Kernel

None - no exogenous variables; F - fixed effect; and T - individual effect and individual trend.

Chapter 18. Forecasting from an Equation

This chapter describes procedures for forecasting and computing fitted values from a single equation. The techniques described here are for forecasting with equation objects estimated using regression methods. Forecasts from equations estimated by specialized techniques, such as ARCH, binary, ordered, tobit, and count methods, are discussed in the corresponding chapters. Forecasting from a series using exponential smoothing methods is explained in "Exponential Smoothing" on page 350, and forecasting using multiple equations and models is described in Chapter 26, "Models", on page 777.

Forecasting from Equations in EViews

To illustrate the process of forecasting from an estimated equation, we begin with a simple example. Suppose we have data on the logarithm of monthly housing starts (HS) and the logarithm of the S&P index (SP) over the period 1959M01–1996M01. The data are contained in a workfile with range 1959M01–1998M12.

We estimate a regression of HS on a constant, SP, and the lag of HS, with an AR(1) to correct for residual serial correlation, using data for the period 1959M01–1990M12, and then use the model to forecast housing starts under a variety of settings. Following estimation, the equation results are held in the equation object EQ01:

Dependent Variable: HS
Method: Least Squares
Date: 01/15/04 Time: 15:57
Sample (adjusted): 1959M03 1990M:01
Included observations: 371 after adjusting endpoints
Convergence achieved after 4 iterations

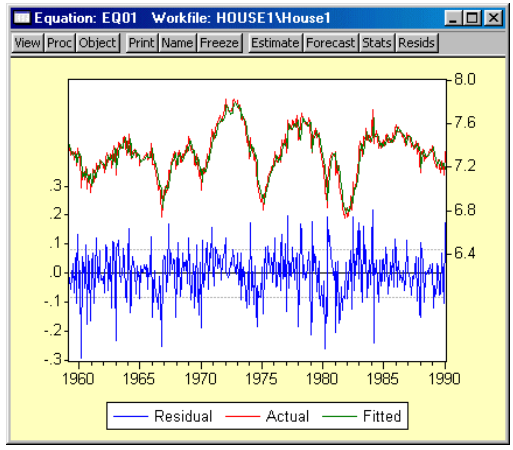
Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.321924	0.117278	2.744975	0.0063
HS(-1)	0.952653	0.016218	58.74157	0.0000
SP	0.005222	0.007588	0.688249	0.4917
AR(1)	-0.271254	0.052114	-5.205027	0.0000

R-squared	0.861373	Mean dependent var	7.324051
Adjusted R-squared	0.860240	S.D. dependent var	0.220996
S.E. of regression	0.082618	Akaike info criterion	-2.138453
Sum squared resid	2.505050	Schwarz criterion	-2.096230
Log likelihood	400.6830	F-statistic	760.1338
Durbin-Watson stat	2.013460	Prob(F-statistic)	0.000000

Inverted AR Roots	-.27

Note that the estimation sample is adjusted by two observations to account for the first difference of the lagged endogenous variable used in deriving AR(1) estimates for this model.

To get a feel for the fit of the model, select **View/Actual, Fitted, Residual…,** then choose **Actual, Fitted, Residual Graph**:

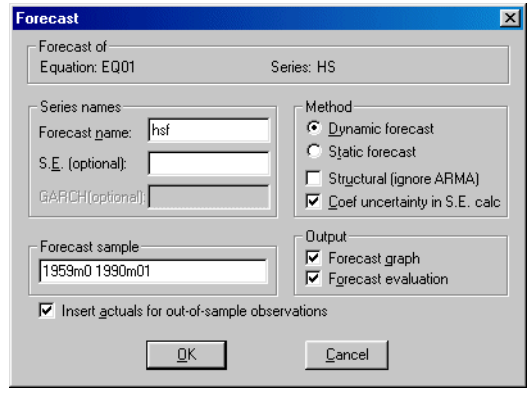


The actual and fitted values depicted on the upper portion of the graph are virtually indistinguishable. This view provides little control over the process of producing fitted values, and does not allow you to save your fitted values. These limitations are overcome by using EViews built-in forecasting procedures to compute fitted values for the dependent variable.

How to Perform a Forecast

To forecast HS from this equation, push the **Forecast** button on the equation toolbar, or select **Proc/Forecast…**.

At the top of the **Forecast** dialog, EViews displays information about the forecast. Here, we show a basic version of the dialog showing that we are forecasting values for the dependent series HS using the estimated EQ01. More complex settings are described in "Forecasting from Equations with Expressions" on page 561.



You should provide the following information:

- **Forecast name**. Fill in the edit box with the series name to be given to your forecast. EViews suggests a name, but you can change it to any valid series name. The name should be different from the name of the dependent variable, since the forecast procedure will overwrite data in the specified series.

- **S.E. (optional)**. If desired, you may provide a name for the series to be filled with the forecast standard errors. If you do not provide a name, no forecast errors will be saved.

- **GARCH (optional)**. For models estimated by ARCH, you will be given a further option of saving forecasts of the conditional variances (GARCH terms). See Chapter 20, "ARCH and GARCH Estimation", on page 601 for a discussion of GARCH estimation.

- **Forecasting method**. You have a choice between **Dynamic** and **Static** forecast methods. **Dynamic** calculates *dynamic, multi-step forecasts* starting from the first period in the forecast sample. In dynamic forecasting, previously forecasted values for the lagged dependent variables are used in forming forecasts of the current value (see "Forecasts with Lagged Dependent Variables" on page 555 and "Forecasting with ARMA Errors" on page 557). This choice will only be available when the estimated equation contains dynamic components, *e.g.*, lagged dependent variables or ARMA terms. **Static** calculates a sequence of *one-step ahead forecasts*, using the actual, rather than forecasted values for lagged dependent variables, if available.

  You may elect to always ignore coefficient uncertainty in computing forecast standard errors (when relevant) by unselecting the **Coef uncertainty in S.E. calc** box.

  In addition, in specifications that contain ARMA terms, you can set the **Structural** option, instructing EViews to ignore any ARMA terms in the equation when forecasting. By default, when your equation has ARMA terms, both dynamic and static solution methods form forecasts of the residuals. If you select **Structural**, all forecasts will ignore the forecasted residuals and will form predictions using only the structural part of the ARMA specification.

- **Sample range**. You must specify the sample to be used for the forecast. By default, EViews sets this sample to be the workfile sample. By specifying a sample outside the sample used in estimating your equation (the *estimation sample*), you can instruct EViews to produce out-of-sample forecasts.

  Note that you are responsible for supplying the values for the independent variables in the out-of-sample forecasting period. For static forecasts, you must also supply the values for any lagged dependent variables.

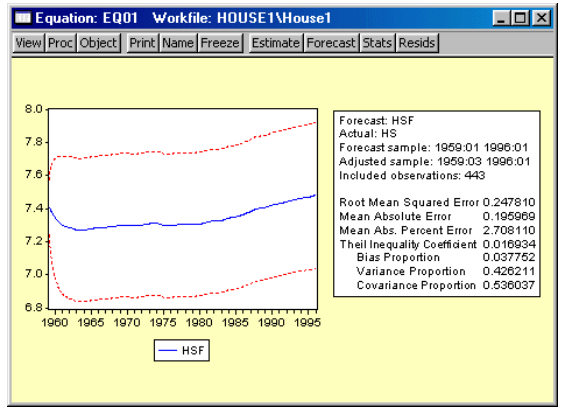
- **Output**. You can choose to see the forecast output as a graph or a numerical forecast evaluation, or both. Forecast evaluation is only available if the forecast sample includes observations for which the dependent variable is observed.

- **Insert actuals for out-of-sample observations**. By default, EViews will fill the forecast series with the values of the actual dependent variable for observations not in the forecast sample. This feature is convenient if you wish to show the divergence of the forecast from the actual values; for observations prior to the beginning of the forecast sample, the two series will contain the same values, then they will diverge as the forecast differs from the actuals. In some contexts, however, you may wish to have forecasted values only for the observations in the forecast sample. If you uncheck this option, EViews will fill the out-of-sample observations with missing values.

Note that when performing forecasts from equations specified using expressions or auto-updating series, you may encounter a version of the **Forecast** dialog that differs from the basic dialog depicted above. See "Forecasting from Equations with Expressions" on page 561 for details.

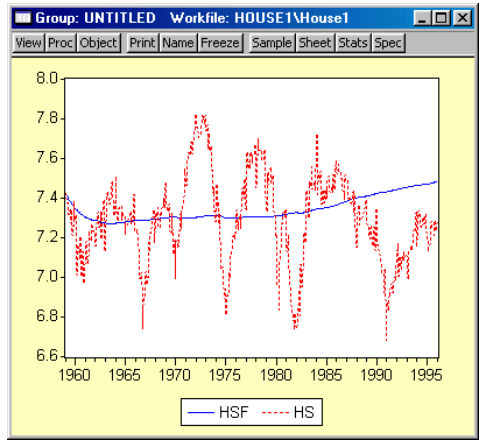
An Illustration

Suppose we produce a dynamic forecast using EQ01 over the sample 1959M01 to 1996M01. The forecast values will be placed in the series HSF, and EViews will display both a graph of the forecasts and the plus and minus two standard error bands, as well as a forecast evaluation:



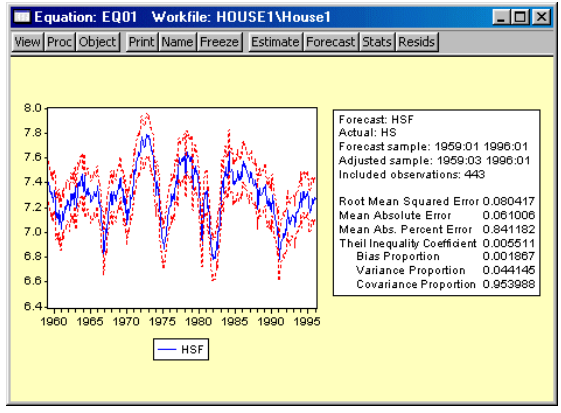
This is a dynamic forecast for the period from 1989M01 through 1996M01. For every period, the previously forecasted values for HS(-1) are used in forming a forecast of the subsequent value of HS. As noted in the output, the forecast values are saved in the series HSF. Since HSF is a standard EViews series, you may examine your forecasts using all of the standard tools for working with series objects.

For example, we may examine the actual versus fitted values by creating a group containing HS and HSF, and plotting the two series. Click on **Quick/Show…** and enter "HS" and "HSF". Then select **View/Graph/Line** to display the two series:

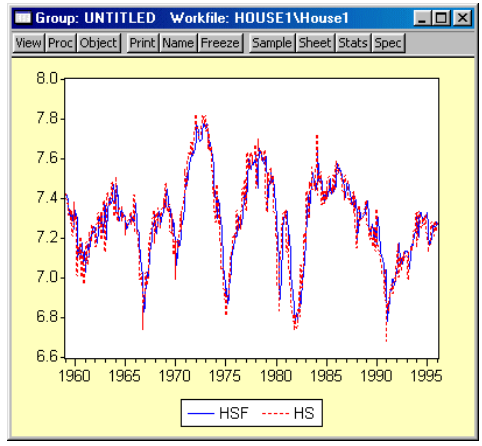


Note the considerable difference between this actual and fitted graph and the **Actual, Fitted, Residual Graph** depicted above.

To perform a series of one-step ahead forecasts, click on **Forecast** on the equation toolbar, and select **Static** forecasts. EViews will display the forecast results:

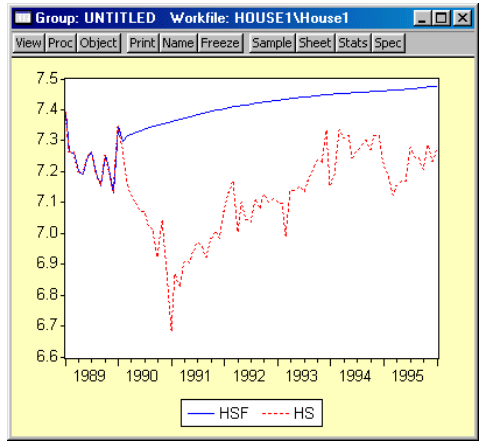


We may also compare the actual and fitted values from the static forecast by examining a line graph of a group containing HS and the new HSF.



The one-step ahead static forecasts are more accurate than the dynamic forecasts since, for each period, the actual value of HS(-1) is used in forming the forecast of HS. These one-step ahead static forecasts are the same forecasts used in the **Actual, Fitted, Residual Graph** displayed above.

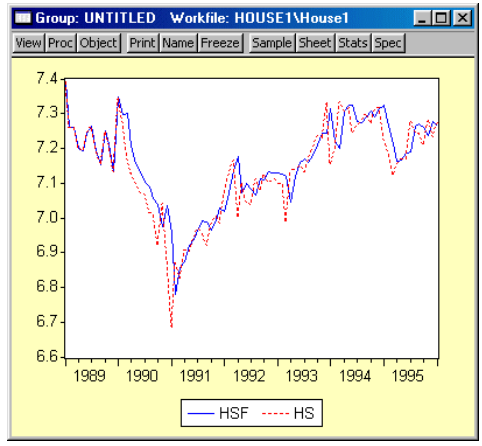
Lastly, we construct a dynamic forecast beginning in 1990M02 (the first period following the estimation sample) and ending in 1996M01. Keep in mind that data are available for SP for this entire period. The plot of the actual and the forecast values for 1989M01 to 1996M01 is given by:



Since we use the default settings for out-of-forecast sample values, EViews backfills the forecast series prior to the forecast sample (up through 1990M01), then dynamically forecasts HS for each subsequent period through 1996M01. This is the forecast that you would

have constructed if, in 1990M01, you predicted values of HS from 1990M02 through 1996M01, given knowledge about the entire path of SP over that period.

The corresponding static forecast is displayed below:



Again, EViews backfills the values of the forecast series, HSF1, through 1990M01. This forecast is the one you would have constructed if, in 1990M01, you used all available data to estimate a model, and then used this estimated model to perform one-step ahead forecasts every month for the next six years.

The remainder of this chapter focuses on the details associated with the construction of these forecasts, the corresponding forecast evaluations, and forecasting in more complex settings involving equations with expressions or auto-updating series.

Forecast Basics

EViews stores the forecast results in the series specified in the **Forecast name** field. We will refer to this series as the *forecast series*.

The *forecast sample* specifies the observations for which EViews will try to compute fitted or forecasted values. If the forecast is not computable, a missing value will be returned. In some cases, EViews will carry out automatic adjustment of the sample to prevent a forecast consisting entirely of missing values (see "Adjustment for Missing Values" on page 550, below). Note that the forecast sample may or may not overlap with the sample of observations used to estimate the equation.

For values not included in the forecast sample, there are two options. By default, EViews fills in the actual values of the dependent variable. If you turn off the **Insert actuals for out-of-sample** option, out-of-forecast-sample values will be filled with NAs.

As a consequence of these rules, *all data in the forecast series will be overwritten during the forecast procedure*. Existing values in the forecast series will be lost.

Computing Point Forecasts

For each observation in the forecast sample, EViews computes the fitted value of the dependent variable using the estimated parameters, the right-hand side exogenous variables, and either the actual or estimated values for lagged endogenous variables and residuals. The method of constructing these forecasted values depends upon the estimated model and user-specified settings.

To illustrate the forecasting procedure, we begin with a simple linear regression model with no lagged endogenous right-hand side variables, and no ARMA terms. Suppose that you have estimated the following equation specification:

```
y c x z
```

Now click on **Forecast**, specify a forecast period, and click **OK**.

For every observation in the forecast period, EViews will compute the fitted value of Y using the estimated parameters and the corresponding values of the regressors, X and Z:

$$\hat{y}_t = \hat{c}(1) + \hat{c}(2)x_t + \hat{c}(3)z_t . \tag{18.1}$$

You should make certain that you have valid values for the exogenous right-hand side variables for all observations in the forecast period. If any data are missing in the forecast sample, the corresponding forecast observation will be an NA.

Adjustment for Missing Values

There are two cases when a missing value will be returned for the forecast value. First, if any of the regressors have a missing value, and second, if any of the regressors are out of the range of the workfile. This includes the implicit error terms in AR models.

In the case of forecasts with no dynamic components in the specification (*i.e.* with no lagged endogenous or ARMA error terms), a missing value in the forecast series will not affect subsequent forecasted values. In the case where there are dynamic components, however, a single missing value in the forecasted series will propagate throughout all future values of the series.

As a convenience feature, EViews will move the starting point of the sample forward where necessary until a valid forecast value is obtained. Without these adjustments, the user would have to figure out the appropriate number of presample values to skip, otherwise the forecast would consist entirely of missing values. For example, suppose you wanted to forecast dynamically from the following equation specification:

```
y c y(-1) ar(1)
```

If you specified the beginning of the forecast sample to the beginning of the workfile range, EViews will adjust forward the forecast sample by 2 observations, and will use the pre-forecast-sample values of the lagged variables (the loss of 2 observations occurs because the residual loses one observation due to the lagged endogenous variable so that the forecast for the error term can begin only from the third observation.)

Forecast Errors and Variances

Suppose the "true" model is given by:

$$y_t = x_t'\beta + \epsilon_t, \tag{18.2}$$

where ϵ_t is an independent, and identically distributed, mean zero random disturbance, and β is a vector of unknown parameters. Below, we relax the restriction that the ϵ's be independent.

The true model generating y is not known, but we obtain estimates b of the unknown parameters β. Then, setting the error term equal to its mean value of zero, the (point) forecasts of y are obtained as:

$$\hat{y}_t = x_t'b. \tag{18.3}$$

Forecasts are made with error, where the error is simply the difference between the actual and forecasted value $e_t = y_t - x_t'b$. Assuming that the model is correctly specified, there are two sources of forecast error: residual uncertainty and coefficient uncertainty.

Residual Uncertainty

The first source of error, termed *residual* or *innovation uncertainty*, arises because the innovations ϵ in the equation are unknown for the forecast period and are replaced with their expectations. While the residuals are zero in expected value, the individual values are non-zero; the larger the variation in the individual errors, the greater the overall error in the forecasts.

The standard measure of this variation is the standard error of the regression (labeled "S.E. of regression" in the equation output). Residual uncertainty is usually the largest source of forecast error.

In dynamic forecasts, innovation uncertainty is compounded by the fact that lagged dependent variables and ARMA terms depend on lagged innovations. EViews also sets these equal to their expected values, which differ randomly from realized values. This additional source of forecast uncertainty tends to rise over the forecast horizon, leading to a pattern of increasing forecast errors. Forecasting with lagged dependent variables and ARMA terms is discussed in more detail below.

Coefficient Uncertainty

The second source of forecast error is *coefficient uncertainty*. The estimated coefficients b of the equation deviate from the true coefficients β in a random fashion. The standard error of the estimated coefficient, given in the regression output, is a measure of the precision with which the estimated coefficients measure the true coefficients.

The effect of coefficient uncertainty depends upon the exogenous variables. Since the estimated coefficients are multiplied by the exogenous variables x in the computation of forecasts, the more the exogenous variables deviate from their mean values, the greater is the forecast uncertainty.

Forecast Variability

The variability of forecasts is measured by the forecast standard errors. For a single equation without lagged dependent variables or ARMA terms, the forecast standard errors are computed as:

$$\text{forecast se} = s\sqrt{1 + x_t'(X'X)^{-1}x_t} \tag{18.4}$$

where s is the standard error of regression. These standard errors account for both innovation (the first term) and coefficient uncertainty (the second term). Point forecasts made from linear regression models estimated by least squares are optimal in the sense that they have the smallest forecast variance among forecasts made by linear unbiased estimators. Moreover, if the innovations are normally distributed, the forecast errors have a t-distribution and forecast intervals can be readily formed.

If you supply a name for the forecast standard errors, EViews computes and saves a series of forecast standard errors in your workfile. You can use these standard errors to form forecast intervals. If you choose the **Do graph** option for output, EViews will plot the forecasts with plus and minus two standard error bands. These two standard error bands provide an approximate 95% forecast interval; if you (hypothetically) make many forecasts, the actual value of the dependent variable will fall inside these bounds 95 percent of the time.

Additional Details

EViews accounts for the additional forecast uncertainty generated when lagged dependent variables are used as explanatory variables (see "Forecasts with Lagged Dependent Variables" on page 555).

There are cases where coefficient uncertainty is ignored in forming the forecast standard error. For example, coefficient uncertainty is always ignored in equations specified by expression, for example, nonlinear least squares, and equations that include PDL (polynomial distributed lag) terms ("Forecasting with Expression and PDL Specifications" on page 567).

In addition, forecast standard errors do not account for GLS weights in estimated panel equations.

Forecast Evaluation

Suppose we construct a dynamic forecast for HS over the period 1990M02 to 1996M01 using our estimated housing equation. If the **Forecast evaluation** option is checked, and there are actual data for the forecasted variable for the forecast sample, EViews reports a table of statistical results evaluating the forecast:

Forecast: HSF
Actual: HS
Sample: 1990M02 1996M01
Include observations: 72

Root Mean Squared Error	0.318700
Mean Absolute Error	0.297261
Mean Absolute Percentage Error	4.205889
Theil Inequality Coefficient	0.021917
Bias Proportion	0.869982
Variance Proportion	0.082804
Covariance Proportion	0.047214

Note that EViews cannot compute a forecast evaluation if there are no data for the dependent variable for the forecast sample.

The forecast evaluation is saved in one of two formats. If you turn on the **Do graph** option, the forecasts are included along with a graph of the forecasts. If you wish to display the evaluations in their own table, you should turn off the **Do graph** option in the Forecast dialog box.

Suppose the forecast sample is $j = T+1, T+2, \ldots, T+h$, and denote the actual and forecasted value in period t as y_t and $\hat{y}_t$, respectively. The reported forecast error statistics are computed as follows:

Root Mean Squared Error	$\sqrt{\sum_{t=T+1}^{T+h} (\hat{y}_t - y_t)^2 / h}$
Mean Absolute Error	$\sum_{t=T+1}^{T+h} \lvert\hat{y}_t - y_t\rvert / h$
Mean Absolute Percentage Error	$100 \sum_{t=T+1}^{T+h} \left\lvert \frac{\hat{y}_t - y_t}{y_t} \right\rvert / h$

Theil Inequality Coefficient	$\frac{\sqrt{\sum_{t=T+1}^{T+h}(\hat{y}_t - y_t)^2/h}}{\sqrt{\sum_{t=T+1}^{T+h}\hat{y}_t^{\,2}/h} + \sqrt{\sum_{t=T+1}^{T+h}y_t^{\,2}/h}}$

The first two forecast error statistics depend on the scale of the dependent variable. These should be used as relative measures to compare forecasts for the same series across different models; the smaller the error, the better the forecasting ability of that model according to that criterion. The remaining two statistics are scale invariant. The Theil inequality coefficient always lies between zero and one, where zero indicates a perfect fit.

The mean squared forecast error can be decomposed as:

$$\sum(\hat{y}_t - y_t)^2/h \;=\; ((\sum\hat{y}_t/h) - \bar{y})^2 + (s_{\hat{y}} - s_y)^2 + 2(1-r)s_{\hat{y}}s_y \qquad (18.5)$$

where $\sum\hat{y}_t/h$, $\bar{y}$, $s_{\hat{y}}$, s_y are the means and (biased) standard deviations of $\hat{y}_t$ and y, and r is the correlation between $\hat{y}$ and y. The proportions are defined as:

Bias Proportion	$\frac{((\sum\hat{y}_t/h) - \bar{y})^2}{\sum(\hat{y}_t - y_t)^2/h}$
Variance Proportion	$\frac{(s_{\bar{y}} - s_y)^2}{\sum(\hat{y}_t - y_t)^2/h}$
Covariance Proportion	$\frac{2(1-r)s_{\bar{y}}s_y}{\sum(\hat{y}_t - y_t)^2/h}$

- The bias proportion tells us how far the mean of the forecast is from the mean of the actual series.
- The variance proportion tells us how far the variation of the forecast is from the variation of the actual series.
- The covariance proportion measures the remaining unsystematic forecasting errors.

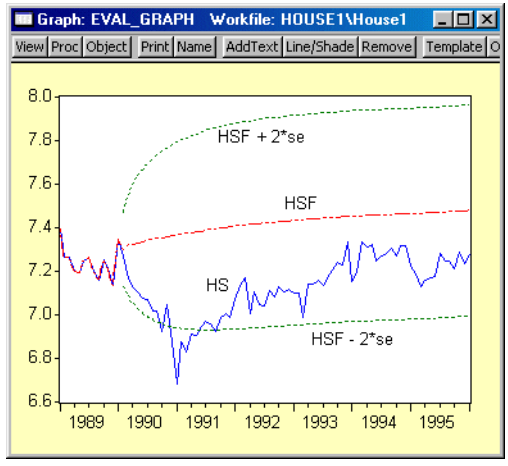
Note that the bias, variance, and covariance proportions add up to one.

If your forecast is "good", the bias and variance proportions should be small so that most of the bias should be concentrated on the covariance proportions. For additional discussion of forecast evaluation, see Pindyck and Rubinfeld (1991, Chapter 12).

For the example output, the bias proportion is large, indicating that the mean of the forecasts does a poor job of tracking the mean of the dependent variable. To check this, we will plot the forecasted series together with the actual series in the forecast sample with the two standard error bounds. Suppose we saved the forecasts and their standard errors as HSF and HSFSE, respectively. Then the plus and minus two standard error series can be generated by the commands:

```
smpl 1990m02 1996m01
series hsf_high = hsf + 2*hsfse
series hsf_low = hsf - 2*hsfse
```

Create a group containing the four series. You can highlight the four series HS, HSF, HSF_HIGH, and HSF_LOW, double click on the selected area, and select **Open Group**, or you can select **Quick/Show...** and enter the four series names. Once you have the group open, select **View/Graph/Line**.



The forecasts completely miss the downturn at the start of the 1990's, but, subsequent to the recovery, track the trend reasonably well from 1992 to 1996.

Forecasts with Lagged Dependent Variables

Forecasting is complicated by the presence of lagged dependent variables on the right-hand side of the equation. For example, we can augment the earlier specification to include the first lag of Y:

```
y c x z y(-1)
```

and click on the **Forecast** button and fill out the series names in the dialog as above. There is some question, however, as to how we should evaluate the lagged value of Y that

appears on the right-hand side of the equation. There are two possibilities: dynamic forecasting and static forecasting.

Dynamic Forecasting

If you select dynamic forecasting, EViews will perform a multi-step forecast of Y, beginning at the start of the forecast sample. For our single lag specification above:

- The initial observation in the forecast sample will use the actual value of lagged Y. Thus, if S is the first observation in the forecast sample, EViews will compute:

$$\hat{y}_S = \hat{c}(1) + \hat{c}(2)x_S + \hat{c}(3)z_S + \hat{c}(4)y_{S-1}, \tag{18.6}$$

  where y_{S-1} is the value of the lagged endogenous variable in the period prior to the start of the forecast sample. This is the one-step ahead forecast.

- Forecasts for subsequent observations will use the previously *forecasted* values of Y:

$$\hat{y}_{S+k} = \hat{c}(1) + \hat{c}(2)x_{S+k} + \hat{c}(3)z_{S+k} + \hat{c}(4)\hat{y}_{S+k-1}. \tag{18.7}$$

- These forecasts may differ significantly from the one-step ahead forecasts.

If there are additional lags of Y in the estimating equation, the above algorithm is modified to account for the non-availability of lagged forecasted values in the additional period. For example, if there are three lags of Y in the equation:

- The first observation (S) uses the actual values for all three lags, y_{S-3}, y_{S-2}, and y_{S-1}.
- The second observation ($S+1$) uses actual values for y_{S-2} and, y_{S-1} and the forecasted value $\hat{y}_S$ of the first lag of y_{S+1}.
- The third observation ($S+2$) will use the actual values for y_{S-1}, and forecasted values $\hat{y}_{S+1}$ and $\hat{y}_S$ for the first and second lags of y_{S+2}.
- All subsequent observations will use the forecasted values for all three lags.

The selection of the start of the forecast sample is very important for dynamic forecasting. The dynamic forecasts are true multi-step forecasts (from the start of the forecast sample), since they use the recursively computed forecast of the lagged value of the dependent variable. These forecasts may be interpreted as the forecasts for subsequent periods that would be computed using information available at the start of the forecast sample.

Dynamic forecasting requires that data for the exogenous variables be available for every observation in the forecast sample, and that values for any lagged dependent variables be observed at the start of the forecast sample (in our example, y_{S-1}, but more generally, any lags of y). If necessary, the forecast sample will be adjusted.

Any missing values for the explanatory variables will generate an NA for that observation and in all subsequent observations, via the dynamic forecasts of the lagged dependent variable.

Static Forecasting

Static forecasting performs a series of one-step ahead forecasts of the dependent variable:

- For each observation in the forecast sample, EViews computes:

$$\hat{y}_{S+k} = \hat{c}(1) + \hat{c}(2)x_{S+k} + \hat{c}(3)z_{S+k} + \hat{c}(4)y_{S+k-1} \tag{18.8}$$

  always using the actual value of the lagged endogenous variable.

Static forecasting requires that data for both the exogenous and any lagged endogenous variables be observed for every observation in the forecast sample. As above, EViews will, if necessary, adjust the forecast sample to account for pre-sample lagged variables. If the data are not available for any period, the forecasted value for that observation will be an NA. The presence of a forecasted value of NA does not have any impact on forecasts for subsequent observations.

A Comparison of Dynamic and Static Forecasting

Both methods will always yield identical results in the first period of a multi-period forecast. Thus, two forecast series, one dynamic and the other static, should be identical for the first observation in the forecast sample.

The two methods will differ for subsequent periods only if there are lagged dependent variables or ARMA terms.

Forecasting with ARMA Errors

Forecasting from equations with ARMA components involves some additional complexities. When you use the AR or MA specifications, you will need to be aware of how EViews handles the forecasts of the lagged residuals which are used in forecasting.

Structural Forecasts

By default, EViews will forecast values for the residuals using the estimated ARMA structure, as described below.

For some types of work, you may wish to assume that the ARMA errors are always zero. If you select the structural forecast option by checking **Structural (ignore ARMA),** EViews computes the forecasts assuming that the errors are always zero. If the equation is estimated without ARMA terms, this option has no effect on the forecasts.

Forecasting with AR Errors

For equations with AR errors, EViews adds forecasts of the residuals from the equation to the forecast of the structural model that is based on the right-hand side variables.

In order to compute an estimate of the residual, EViews requires estimates or actual values of the lagged residuals. For the first observation in the forecast sample, EViews will use pre-sample data to compute the lagged residuals. If the pre-sample data needed to compute the lagged residuals are not available, EViews will adjust the forecast sample, and backfill the forecast series with actual values (see the discussion of "Adjustment for Missing Values" on page 550).

If you choose the **Dynamic** option, both the lagged dependent variable and the lagged residuals will be forecasted dynamically. If you select **Static**, both will be set to the actual lagged values. For example, consider the following AR(2) model:

$$\begin{aligned} y_t &= x_t'\beta + u_t \\ u_t &= \rho_1 u_{t-1} + \rho_2 u_{t-2} + \epsilon_t \end{aligned} \tag{18.9}$$

Denote the fitted residuals as $e_t = y_t - x_t'b$, and suppose the model was estimated using data up to $t = S-1$. Then, provided that the x_t values are available, the static and dynamic forecasts for $t = S, S+1, \ldots$, are given by:

	Static	**Dynamic**
$\hat{y}_S$	$x_S'b + \hat{\rho}_1 e_{S-1} + \hat{\rho}_2 e_{S-2}$	$x_S'b + \hat{\rho}_1 e_{S-1} + \hat{\rho}_2 e_{S-2}$
$\hat{y}_{S+1}$	$x_{S+1}'b + \hat{\rho}_1 e_S + \hat{\rho}_2 e_{S-1}$	$x_{S+1}'b + \hat{\rho}_1 \hat{u}_S + \hat{\rho}_2 e_{S-1}$
$\hat{y}_{S+2}$	$x_{S+2}'b + \hat{\rho}_1 e_{S+1} + \hat{\rho}_2 e_S$	$x_{S+2}'b + \hat{\rho}_1 \hat{u}_{S+1} + \hat{\rho}_2 \hat{u}_S$

where the residuals $\hat{u}_t = \hat{y}_t - x_t'b$ are formed using the forecasted values of y_t. For subsequent observations, the dynamic forecast will always use the residuals based upon the multi-step forecasts, while the static forecast will use the one-step ahead forecast residuals.

Forecasting with MA Errors

In general, you need not concern yourselves with the details of MA forecasting, since EViews will do all of the work for you. For those of you who are interested in the details of dynamic forecasting, however, the following discussion should aid you in relating EViews results with those obtained from other sources.

The first step in computing forecasts using MA terms is to obtain fitted values for the innovations in the pre-forecast sample period. For example, if you are forecasting the values of y, beginning in period S, with a simple MA(q):

$$\hat{y}_S = \epsilon_S + \hat{\phi}_1 \epsilon_{S-1} + \ldots + \hat{\phi}_q \epsilon_{S-q}, \tag{18.10}$$

you will need values for the lagged innovations, $\epsilon_{S-1}, \epsilon_{S-2}, \ldots, \epsilon_{S-q}$.

To compute these pre-forecast innovations, EViews will first assign values for the q innovations prior to the start of the *estimation* sample, $\epsilon_0, \epsilon_{-1}, \epsilon_{-2}, \ldots, \epsilon_{-q}$. If your equation is estimated with backcasting turned on, EViews will perform backcasting to obtain these values. If your equation is estimated with backcasting turned off, or if the forecast sample precedes the estimation sample, the initial values will be set to zero.

Given the initial values, EViews will fit the values of subsequent innovations, $\epsilon_1, \epsilon_2, \ldots, \epsilon_q, \ldots, \epsilon_{S-1}$, using forward recursion. The backcasting and recursion procedures are described in detail in the discussion of backcasting in ARMA models in "Backcasting MA terms" on page 510.

Note the difference between this procedure and the approach for AR errors outlined above, in which the forecast sample is adjusted forward and the pre-forecast values are set to actual values.

The choice between dynamic and static forecasting has two primary implications:

- Once the q pre-sample values for the innovations are computed, dynamic forecasting sets subsequent innovations to zero. Static forecasting extends the forward recursion through the end of the estimation sample, allowing for a series of one-step ahead forecasts of both the structural model and the innovations.
- When computing static forecasts, EViews uses the entire estimation sample to backcast the innovations. For dynamic MA forecasting, the backcasting procedure uses observations from the beginning of the estimation sample to either the beginning of the forecast period, or the end of the estimation sample, whichever comes first.

Example

As an example of forecasting from ARMA models, consider forecasting the monthly new housing starts (HS) series. The estimation period is 1959M01–1984M12 and we forecast for the period 1985M01–1991M12. We estimated the following simple multiplicative seasonal autoregressive model,

```
hs c ar(1) sar(12)
```

yielding:

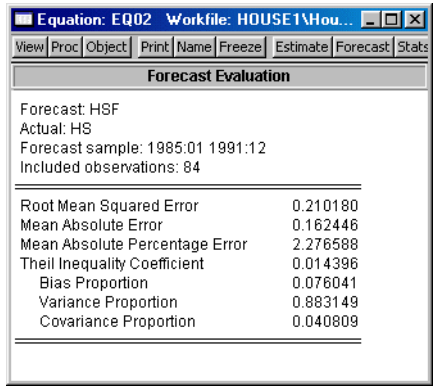
Dependent Variable: HS
Method: Least Squares
Date: 01/15/04 Time: 16:34
Sample (adjusted): 1960M02 1984M12
Included observations: 299 after adjusting endpoints
Convergence achieved after 5 iterations

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	7.317283	0.071371	102.5243	0.0000
AR(1)	0.935392	0.021028	44.48403	0.0000
SAR(12)	-0.113868	0.060510	-1.881798	0.0608

R-squared	0.862967	Mean dependent var	7.313496
Adjusted R-squared	0.862041	S.D. dependent var	0.239053
S.E. of regression	0.088791	Akaike info criterion	-1.995080
Sum squared resid	2.333617	Schwarz criterion	-1.957952
Log likelihood	301.2645	F-statistic	932.0312
Durbin-Watson stat	2.452568	Prob(F-statistic)	0.000000

Inverted AR Roots	.94	.81+.22i	.81-.22i	.59+.59i
	.59-.59i	.22-.81i	.22+.81i	-.22+.81i
	-.22-.81i	-.59+.59i	-.59-.59i	-.81-.22i
	-.81+.22i			

To perform a dynamic forecast from this estimated model, click **Forecast** on the equation toolbar, select **Forecast evaluation** and unselect **Forecast graph**. The forecast evaluation statistics for the model are shown below:

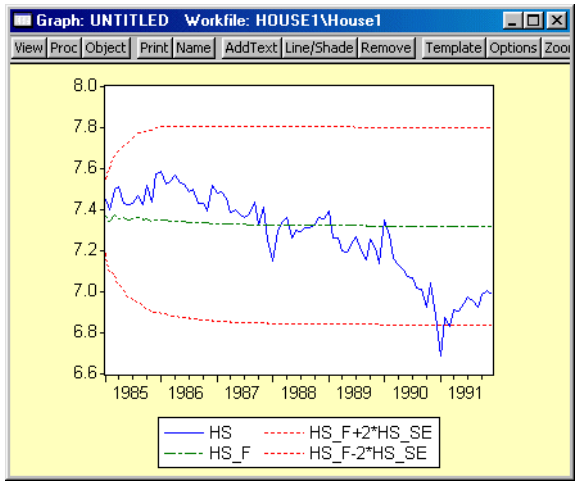


The large variance proportion indicates that the forecasts are not tracking the variation in the actual HS series. To plot the actual and forecasted series together with the two standard error bands, you can type:

```
smpl 1985m01 1991m12
```

```
plot hs hs_f hs_f+2*hs_se hs_f-2*hs_se
```

where HS_F and HS_SE are the forecasts and standard errors of HS.



As indicated by the large variance proportion, the forecasts track the seasonal movements in HS only at the beginning of the forecast sample and quickly flatten out to the mean forecast value.

Forecasting from Equations with Expressions

One of the most useful EViews innovations is the ability to estimate and forecast from equations that are specified using expressions or auto-updating series. You may, for example, specify your dependent variable as LOG(X), or use an auto-updating regressor series EXPZ that is defined using the expression EXP(Z). Using expressions or auto-updating series in equations creates no added complexity for estimation since EViews simply evaluates the implicit series prior to computing the equation estimator.

The use of expressions in equations does raise issues when computing forecasts from equations. While not particularly complex or difficult to address, the situation does require a basic understanding of the issues involved, and some care must be taken when specifying your forecast.

In discussing the relevant issues, we distinguish between specifications that contain only auto-series expressions such as LOG(X), and those that contain auto-updating series such as EXPZ.

Forecasting using Auto-series Expressions

When forecasting from an equation that contains only ordinary series or auto-series expressions such as LOG(X), issues arise only when the dependent variable is specified using an expression.

Point Forecasts

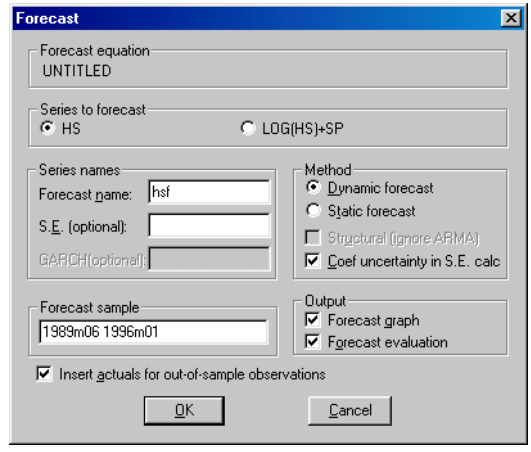
EViews always provides you with the option to forecast the dependent variable expression. If the expression can be normalized (solved for the first series in the expression), EViews also provides you with the option to forecast the normalized series.

For example, suppose you estimated an equation with the specification:

```
(log(hs)+sp) c hs(-1)
```

If you press the **Forecast** button, EViews will open a dialog prompting you for your forecast specification.

The resulting **Forecast** dialog is a slightly more complex version of the basic dialog, providing you with a new section allowing you to choose between two series to forecast: the normalized series, HS, or the equation dependent variable, LOG(HS) + SP.



Simply select the radio button for the desired forecast series. Note that you are not provided with the opportunity to forecast SP directly since HS, the first series that appears on the left-hand side of the estimation equation, is offered as the choice of normalized series.

It is important to note that the **Dynamic forecast** method is available since EViews is able to determine that the forecast equation has dynamic elements, with HS appearing on the left-hand side of the equation (either directly as HS or in the expression LOG(HS) + SP) and on the right-hand side of the equation in lagged form. If you select dynamic forecasting, previously forecasted values for HS(-1) will be used in forming forecasts of either HS or LOG(HS) + SP.

If the formula can be normalized, EViews will compute the forecasts of the transformed dependent variable by first forecasting the normalized series and then transforming the

forecasts of the normalized series. This methodology has important consequences when the formula includes lagged series. For example, consider the following two models:

```
series dhs = d(hs)
equation eq1.ls d(hs) c sp
equation eq2.ls dhs c sp
```

The dynamic forecasts of the first difference D(HS) from the first equation will be numerically identical to those for DHS from the second equation. However, the static forecasts for D(HS) from the two equations will not be identical. In the first equation, EViews knows that the dependent variable is a transformation of HS, so it will use the actual lagged value of HS in computing the static forecast of the first difference D(HS). In the second equation, EViews simply views DY as an ordinary series, so that only the estimated constant and SP are used to compute the static forecast.

One additional word of caution–when you have dependent variables that use lagged values of a series, you should avoid referring to the lagged series before the current series in a dependent variable expression. For example, consider the two equation specifications:

```
d(hs) c sp
(-hs(-1)+hs) c sp
```

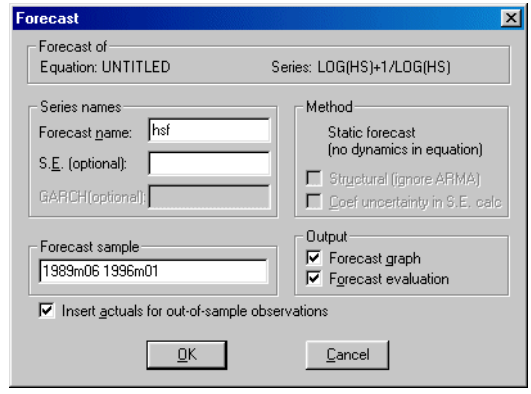
Both specifications have the first difference of HS as the dependent variable and the estimation results are identical for the two models. However, if you forecast HS from the second model, EViews will try to calculate the forecasts of HS using leads of the actual series HS. These forecasts of HS will differ from those produced by the first model, which may not be what you expected.

In some cases, EViews will not be able to normalize the dependent variable expression. In this case, the **Forecast** dialog will only offer you the option of forecasting the entire expression. If, for example, you specify your equation as:

```
log(hs)+1/log(hs) = c(1) + c(2)*hs(-1)
```

EViews will not be able to normalize the dependent variable for forecasting. The corresponding **Forecast** dialog will reflect this fact.

This version of the dialog only allows you to forecast the dependent variable expression, since EViews is unable to normalize and solve for HS. Note also that only static forecasts are available for this case since EViews is unable to solve for lagged values of HS on the right hand-side.



Plotted Standard Errors

When you select **Do graph** in the forecast dialog, EViews will plot the forecasts, along with plus and minus two standard error bands. When you estimate an equation with an expression for the left-hand side, EViews will plot the standard error bands for either the normalized or the unnormalized expression, depending upon which term you elect to forecast.

If you elect to predict the normalized dependent variable, EViews will automatically account for any nonlinearity in the standard error transformation. The next section provides additional details on the procedure used to normalize the upper and lower error bounds.

Saved Forecast Standard Errors

If you provide a name in this edit box, EViews will store the standard errors of the underlying series or expression that you chose to forecast.

When the dependent variable of the equation is a simple series or an expression involving only *linear* transformations, the saved standard errors will be exact (except where the forecasts do not account for coefficient uncertainty, as described below). If the dependent variable involves nonlinear transformations, the saved forecast standard errors will be exact if you choose to forecast the entire formula. If you choose to forecast the underlying endogenous series, the forecast uncertainty cannot be computed exactly, and EViews will provide a linear (first-order) approximation to the forecast standard errors.

Consider the following equations involving a formula dependent variable:

```
d(hs) c sp
log(hs) c sp
```

For the first equation, you may choose to forecast either HS or D(HS). In both cases, the forecast standard errors will be exact, since the expression involves only linear transformations. The two standard errors will, however, differ in dynamic forecasts since the forecast standard errors for HS take into account the forecast uncertainty from the lagged value of

HS. In the second example, the forecast standard errors for LOG(HS) will be exact. If, however, you request a forecast for HS itself, the standard errors saved in the series will be the approximate (linearized) forecast standard errors for HS.

Note that when EViews displays a graph view of the forecasts together with standard error bands, the standard error bands are always exact. Thus, in forecasting the underlying dependent variable in a nonlinear expression, the standard error bands will not be the same as those you would obtain by constructing series using the linearized standard errors saved in the workfile.

Suppose in our second example above that you store the forecast of HS and its standard errors in the workfile as the series HSHAT and SE_HSHAT. Then the *approximate* two standard error bounds can be generated manually as:

```
series hshat_high1 = hshat + 2*se_hshat
series hshat_low1 = hshat - 2*se_hshat
```

These forecast error bounds will be symmetric about the point forecasts HSHAT.

On the other hand, when EViews plots the forecast error bounds of HS, it proceeds in two steps. It first obtains the forecast of LOG(HS) and its standard errors (named, say, LHSHAT and SE_LHSHAT) and forms the forecast error bounds on LOG(HS):

```
lhshat + 2*se_lhshat
lhshat - 2*se_lhshat
```

It then normalizes (inverts the transformation) of the two standard error bounds to obtain the prediction interval for HS:

```
series hshat_high2 = exp(hshat + 2*se_hshat)
series hshat_low2 = exp(hshat - 2*se_hshat)
```

Because this transformation is a non-linear transformation, these bands will not be symmetric around the forecast.

To take a more complicated example, suppose that you generate the series DLHS and LHS, and then estimate three equivalent models:

```
series dlhs = dlog(hs)
series lhs = log(hs)
equation eq1.ls dlog(hs) c sp
equation eq2.ls d(lhs) c sp
equation eq3.ls dlhs c sp
```

The estimated equations from the three models are numerically identical. If you choose to forecast the underlying dependent (normalized) series from each model, EQ1 will forecast HS, EQ2 will forecast LHS (the log of HS), and EQ3 will forecast DLHS (the first difference

of the logs of HS, LOG(HS)–LOG(HS(–1)). The forecast standard errors saved from EQ1 will be linearized approximations to the forecast standard error of HS, while those from the latter two will be exact for the forecast standard error of LOG(HS) and the first difference of the logs of HS.

Static forecasts from all three models are identical because the forecasts from previous periods are not used in calculating this period's forecast when performing static forecasts. For dynamic forecasts, the log of the forecasts from EQ1 will be identical to those from EQ2 and the log first difference of the forecasts from EQ1 will be identical to the first difference of the forecasts from EQ2 and to the forecasts from EQ3. For static forecasts, the log first difference of the forecasts from EQ1 will be identical to the first difference of the forecasts from EQ2. However, these forecasts differ from those obtained from EQ3 because EViews does not know that the generated series DLY is actually a difference term so that it does not use the dynamic relation in the forecasts.

Forecasting with Auto-updating series

When forecasting from an equation that contains auto-updating series defined by formulae, the central question is whether EViews interprets the series as ordinary series, or whether it treats the auto-updating series as expressions.

Suppose for example, that we have defined auto-updating series LOGHS and LOGHSLAG, for the log of HAS and the log of HS(-1), respectively,

```
frml loghs = log(hs)
frml loghslag = log(hs(-1))
```

and that we employ these auto-updating series in estimating an equation specification:

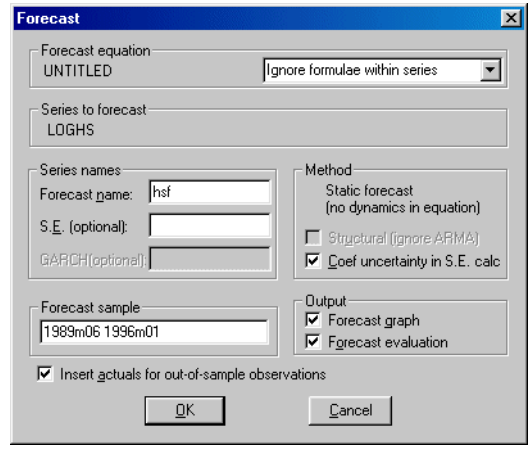
```
loghs c loghslag
```

It is worth pointing out this specification yields results that are identical to those obtained from estimating an equation using the expressions directly using LOG(HS) and LOG(HS(-1)):

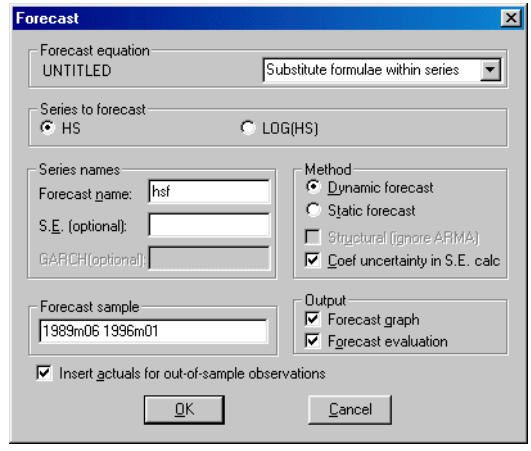
```
log(hs) c log(hs(-1))
```

The **Forecast** dialog for the first equation specification (using LOGHS and LOGHSLAG) contains an additional combo box allowing you to specify whether to interpret the auto-updating series as ordinary series, or whether to look inside LOGHS and LOGHSLAG to use their expressions.

By default, the combo box is set to **Ignore formulae within series**, so that LOGHS and LOGHSLAG are viewed as ordinary series. Note that since EViews ignores the expressions underlying the auto-updating series, you may only forecast the dependent series LOGHS, and there are no dynamics implied by the equation.



Alternatively, you may instruct EViews to use the expressions in place of all auto-updating series by changing the combo box setting to **Substitute formulae within series**.



If you elect to substitute the formulae, the **Forecast** dialog will change to reflect the use of the underlying expressions as you may now choose between forecasting HS or LOG(HS). We also see that when you use the substituted expressions you are able to perform either dynamic or static forecasting.

It is worth noting that substituting expressions yields a **Forecast** dialog that offers the same options as if you were to forecast from the second equation specification above—using LOG(HS) as the dependent series expression, and LOG(HS(-1)) as an independent series expression.

Forecasting with Expression and PDL Specifications

As explained above, forecast errors can arise from two sources: coefficient uncertainty and innovation uncertainty. For linear regression models, the forecast standard errors account for both coefficient and innovation uncertainty. However, if the model is specified by expression (or if it contains a PDL specification), then the standard errors ignore coefficient uncertainty. EViews will display a message in the status line at the bottom of the EViews window when forecast standard errors only account for innovation uncertainty.

For example, consider the three specifications:

```
log(y) c x
y = c(1) + c(2)*x
y = exp(c(1)*x)
y c x pdl(z, 4, 2)
```

Forecast standard errors from the first model account for both coefficient and innovation uncertainty since the model is specified by list, and does not contain a PDL specification. The remaining specifications have forecast standard errors that account only for residual uncertainty.

Chapter 19. Specification and Diagnostic Tests

Empirical research is usually an interactive process. The process begins with a specification of the relationship to be estimated. Selecting a specification usually involves several choices: the variables to be included, the functional form connecting these variables, and if the data are time series, the dynamic structure of the relationship between the variables.

Inevitably, there is uncertainty regarding the appropriateness of this initial specification. Once you estimate your equation, EViews provides tools for evaluating the quality of your specification along a number of dimensions. In turn, the results of these tests influence the chosen specification, and the process is repeated.

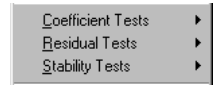
This chapter describes the extensive menu of specification test statistics that are available as views or procedures of an equation object. While we attempt to provide you with sufficient statistical background to conduct the tests, practical considerations ensure that many of the descriptions are incomplete. We refer you to standard statistical and econometric references for further details.

Background

Each test procedure described below involves the specification of a null hypothesis, which is the hypothesis under test. Output from a test command consists of the sample values of one or more test statistics and their associated probability numbers (p-values). The latter indicate the probability of obtaining a test statistic whose absolute value is greater than or equal to that of the sample statistic if the null hypothesis is true. Thus, low p-values lead to the rejection of the null hypothesis. For example, if a p-value lies between 0.05 and 0.01, the null hypothesis is rejected at the 5 percent but not at the 1 percent level.

Bear in mind that there are different assumptions and distributional results associated with each test. For example, some of the test statistics have exact, finite sample distributions (usually t or F-distributions). Others are large sample test statistics with asymptotic χ^2 distributions. Details vary from one test to another and are given below in the description of each test.

The **View** button on the equation toolbar gives you a choice among three categories of tests to check the specification of the equation.

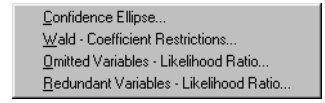


Additional tests are discussed elsewhere in the *User's Guide*. These tests include unit root tests ("Performing Unit Root Tests in EViews" on page 518), the Granger causality test ("Granger Causality" on page 388), tests specific to binary, order, censored, and count models (Chapter 21, "Discrete and Limited Dependent Variable Models", on page 621), and the Johansen test for cointegration ("How to Perform a Cointegration Test" on page 740).

Coefficient Tests

These tests evaluate restrictions on the estimated coefficients, including the special case of tests for omitted and redundant variables.

Confidence Ellipses


Confidence Ellipse...
Wald - Coefficient Restrictions...
Omitted Variables - Likelihood Ratio...
Redundant Variables - Likelihood Ratio...

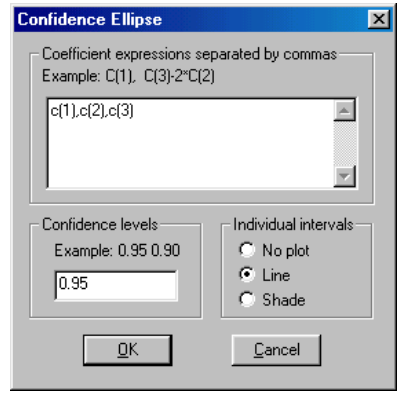
The confidence ellipse view plots the joint confidence region of any two functions of estimated parameters from an EViews estimation object. Along with the ellipses, you can choose to display the individual confidence intervals.

We motivate our discussion of this view by pointing out that the Wald test view (**View/Coefficient Tests/Wald - Coefficient Restrictions...**) allows you to test restrictions on the estimated coefficients from an estimation object. When you perform a Wald test, EViews provides a table of output showing the numeric values associated with the test.

An alternative approach to displaying the results of a Wald test is to display a confidence interval. For a given test size, say 5%, we may display the one-dimensional interval within which the test statistic must lie for us not to reject the null hypothesis. Comparing the realization of the test statistic to the interval corresponds to performing the Wald test.

The one-dimensional confidence interval may be generalized to the case involving two restrictions, where we form a joint confidence region, or confidence ellipse. The confidence ellipse may be interpreted as the region in which the realization of two test statistics must lie for us not to reject the null.

To display confidence ellipses in EViews, simply select **View/Coefficient Tests/Confidence Ellipse...** from the estimation object toolbar. EViews will display a dialog prompting you to specify the coefficient restrictions and test size, and to select display options.



The first part of the dialog is identical to that found in the Wald test view—here, you will enter your coefficient restrictions into the edit box, with multiple restrictions separated by commas. The computation of the confidence ellipse requires a minimum of two restrictions. If you provide more than two restrictions, EViews will display all unique pairs of confidence ellipses.

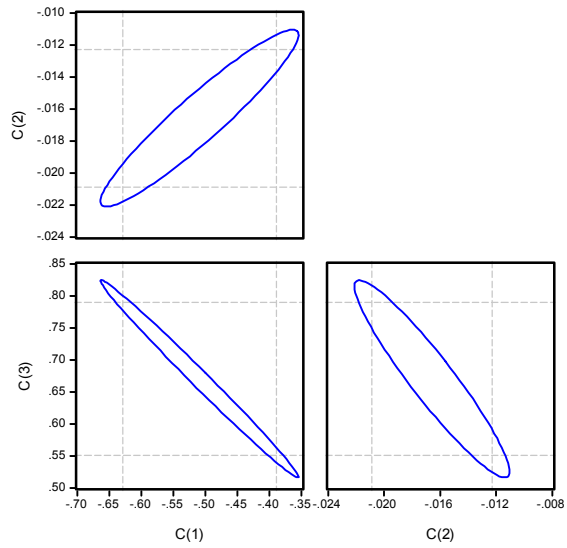
In this simple example depicted here, we provide a (comma separated) list of coefficients from the estimated equation. This description of the restrictions takes advantage of the fact that EViews interprets any expression without an explicit equal sign as being equal to zero (so that "C(1)" and

"C(1) = 0" are equivalent). You may, of course, enter an explicit restriction involving an equal sign (for example, "C(1) + C(2) = C(3)/2").

Next, select a size or sizes for the confidence ellipses. Here, we instruct EViews to construct a 95% confidence ellipse. Under the null hypothesis, the test statistic values will fall outside of the corresponding confidence ellipse 5% of the time.

Lastly, we choose a display option for the individual confidence intervals. If you select **Line** or **Shade**, EViews will mark the confidence interval for each restriction, allowing you to see, at a glance, the individual results. **Line** will display the individual confidence intervals as dotted lines; **Shade** will display the confidence intervals as a shaded region. If you select **None**, EViews will not display the individual intervals.

The output depicts three confidence ellipses that result from pairwise tests implied by the three restrictions ("C(1) = 0", "C(2) = 0", and "C(3) = 0").

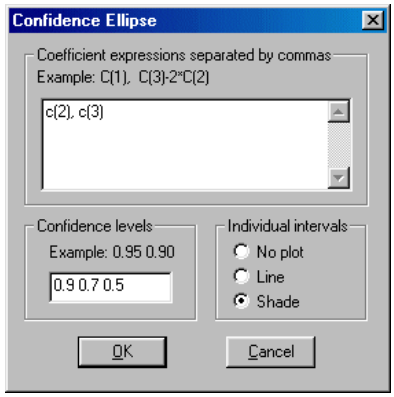


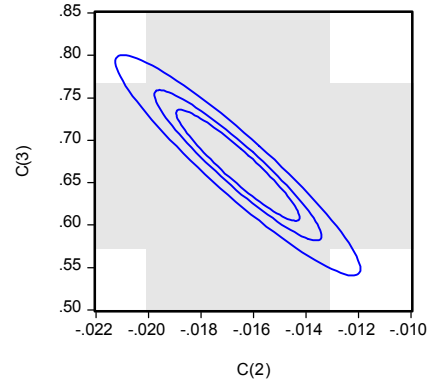
Notice first the presence of the dotted lines showing the corresponding confidence intervals for the individual coefficients.

The next thing that jumps out from this example is that the coefficient estimates are highly correlated—if the estimates were independent, the ellipses would be exact circles.

You can easily see the importance of this correlation. For example, focusing on the ellipse for C(1) and C(3) depicted in the lower left-hand corner, an estimated C(1) of –.65 is sufficient reject the hypothesis that C(1) = 0 (since it falls below the end of the univariate confidence interval). If C(3) = .8, we cannot reject the joint null that C(1) = 0, and C(3) = 0 (since C(1) = -.65, C(3) = .8 falls within the confidence ellipse).

EViews allows you to display more than one size for your confidence ellipses. This feature allows you to draw confidence contours so that you may see how the rejection region changes at different probability values. To do so, simply enter a space delimited list of confidence levels. Note that while the coefficient restriction expressions must be separated by commas, the contour levels must be separated by spaces.





Here, the individual confidence intervals are depicted with shading. The individual intervals are based on the largest size confidence level (which has the widest interval), in this case, 0.9.

Computational Details

Consider two functions of the parameters $f_1(\beta)$ and $f_2(\beta)$, and define the bivariate function $f(\beta) = (f_1(\beta), f_2(\beta))$.

The size α joint confidence ellipse is defined as the set of points b such that:

$$(b - f(\hat{\beta}))'(V(\hat{\beta})^{-1})(b - f(\hat{\beta})) = c_\alpha \tag{19.1}$$

where $\hat{\beta}$ are the parameter estimates, $V(\hat{\beta})$ is the covariance matrix of $\hat{\beta}$, and c_α is the size α critical value for the related distribution. If the parameter estimates are least-squares based, the $F(2, n-2)$ distribution is used; if the parameter estimates are likelihood based, the $\chi^2(2)$ distribution will be employed.

The individual intervals are two-sided intervals based on either the *t*-distribution (in the cases where c_α is computed using the *F*-distribution), or the normal distribution (where c_α is taken from the χ^2 distribution).

Wald Test (Coefficient Restrictions)

The Wald test computes a test statistic based on the unrestricted regression. The Wald statistic measures how close the unrestricted estimates come to satisfying the restrictions under the null hypothesis. If the restrictions are in fact true, then the unrestricted estimates should come close to satisfying the restrictions.

How to Perform Wald Coefficient Tests

To demonstrate the calculation of Wald tests in EViews, we consider simple examples. Suppose a Cobb-Douglas production function has been estimated in the form:

$$\log Q = A + \alpha \log L + \beta \log K + \epsilon \,, \tag{19.2}$$

where Q, K and L denote value-added output and the inputs of capital and labor respectively. The hypothesis of constant returns to scale is then tested by the restriction: $\alpha + \beta = 1$.

Estimation of the Cobb-Douglas production function using annual data from 1947 to 1971 provided the following result:

Dependent Variable: LOG(Q)
Method: Least Squares
Date: 08/11/97 Time: 16:56
Sample: 1947 1971
Included observations: 25

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-2.327939	0.410601	-5.669595	0.0000
LOG(L)	1.591175	0.167740	9.485970	0.0000
LOG(K)	0.239604	0.105390	2.273498	0.0331

R-squared	0.983672	Mean dependent var	4.767586
Adjusted R-squared	0.982187	S.D. dependent var	0.326086
S.E. of regression	0.043521	Akaike info criterion	-3.318997
Sum squared resid	0.041669	Schwarz criterion	-3.172732
Log likelihood	44.48746	F-statistic	662.6819
Durbin-Watson stat	0.637300	Prob(F-statistic)	0.000000

The sum of the coefficients on LOG(L) and LOG(K) appears to be in excess of one, but to determine whether the difference is statistically relevant, we will conduct the hypothesis test of constant returns.

To carry out a Wald test, choose **View/Coefficient Tests/Wald-Coefficient Restrictions...** from the equation toolbar. Enter the restrictions into the edit box, with multiple coefficient restrictions separated by commas. The restrictions should be expressed as equations involving the estimated coefficients and constants. The coefficients should be referred to as C(1), C(2), and so on, unless you have used a different coefficient vector in estimation.

If you enter a restriction that involves a series name, EViews will prompt you to enter an observation at which the test statistic will be evaluated. The value of the series will at that period will be treated as a constant for purposes of constructing the test statistic.

To test the hypothesis of constant returns to scale, type the following restriction in the dialog box:

```
c(2) + c(3) = 1
```

and click **OK**. EViews reports the following result of the Wald test:

Wald Test:
Equation: EQ1

Test Statistic	Value	df	Probability
Chi-square	120.0177	1	0.0000
F-statistic	120.0177	(1, 22)	0.0000

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
-1 + C(2) + C(3)	0.830779	0.075834

Restrictions are linear in coefficients.

EViews reports an F-statistic and a Chi-square statistic with associated p-values. See "Wald Test Details" on page 576 for a discussion of these statistics. In addition, EViews reports the value of the normalized (homogeneous) restriction and an associated standard error. In this example, we have a single linear restriction so the two test statistics are identical, with the p-value indicating that we can decisively reject the null hypothesis of constant returns to scale.

To test more than one restriction, separate the restrictions by commas. For example, to test the hypothesis that the elasticity of output with respect to labor is 2/3 and the elasticity with respect to capital is 1/3, enter the restrictions as,

```
c(2)=2/3, c(3)=1/3
```

and EViews reports:

Wald Test:
Equation: EQ1

Test Statistic	Value	df	Probability
Chi-square	53.99105	2	0.0000
F-statistic	26.99553	(2, 22)	0.0000

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
-2/3 + C(2)	0.924508	0.167740
-1/3 + C(1)	-2.661272	0.410601

Restrictions are linear in coefficients.

Note that in addition to the test statistic summary, we report the values of both of the normalized restrictions, along with their standard errors (the square roots of the diagonal elements of the restriction covariance matrix).

As an example of a nonlinear model with a nonlinear restriction, we estimate a production function of the form:

$$\log Q = \beta_1 + \beta_2 \log(\beta_3 K^{\beta_4} + (1 - \beta_3) L^{\beta_4}) + \epsilon \tag{19.3}$$

and test the constant elasticity of substitution (CES) production function restriction $\beta_2 = 1/\beta_4$. This is an example of a nonlinear restriction. To estimate the (unrestricted) nonlinear model, you should select **Quick/Estimate Equation…** and then enter the following specification:

```
log(q) = c(1) + c(2)*log(c(3)*k^c(4)+(1-c(3))*l^c(4))
```

To test the nonlinear restriction, choose **View/Coefficient Tests/Wald-Coefficient Restrictions…** from the equation toolbar and type the following restriction in the Wald Test dialog box:

```
c(2)=1/c(4)
```

The results are presented below:

Wald Test:
Equation: EQ2

Test Statistic	Value	df	Probability
Chi-square	0.028508	1	0.8659
F-statistic	0.028508	(1, 21)	0.8675

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
C(2) - 1/C(4)	1.292163	7.653088

Delta method computed using analytic derivatives.

Since this is a nonlinear equation, we focus on the Chi-square statistic which fails to reject the null hypothesis. Note that EViews reports that it used the delta method (with analytic derivatives) to compute the Wald restriction variance for the nonlinear restriction.

It is well-known that nonlinear Wald tests are not invariant to the way that you specify the nonlinear restrictions. In this example, the nonlinear restriction $\beta_2 = 1/\beta_4$ may equivalently be written as $\beta_2\beta_4 = 1$ or $\beta_4 = 1/\beta_2$ (for nonzero β_2 and β_4). For example, entering the restriction as,

```
c(2)*c(4)=1
```

yields:

Wald Test:
Equation: EQ2

Test Statistic	Value	df	Probability
Chi-square	104.5599	1	0.0000
F-statistic	104.5599	(1, 21)	0.0000

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
-1 + C(2)*C(4)	0.835330	0.081691

Delta method computed using analytic derivatives.

so that the test now decisively rejects the null hypothesis. We hasten to add that type of inconsistency is not unique to EViews, but is a more general property of the Wald test. Unfortunately, there does not seem to be a general solution to this problem (see Davidson and MacKinnon, 1993, Chapter 13).

Wald Test Details

Consider a general nonlinear regression model:

$$y = f(\beta) + \epsilon \tag{19.4}$$

where y and ϵ are T-vectors and β is a k-vector of parameters to be estimated. Any restrictions on the parameters can be written as:

$$H_0\colon\ g(\beta) = 0\,, \tag{19.5}$$

where g is a smooth function, $g\colon R^k \rightarrow R^q$, imposing q restrictions on β. The Wald statistic is then computed as:

$$W = g(\beta)'\Big(\frac{\partial g(\beta)}{\partial \beta}\hat{V}(b)\frac{\partial g(\beta)}{\partial \beta'}\Big)g(\beta)\big|_{\beta = b} \tag{19.6}$$

where T is the number of observations and b is the vector of unrestricted parameter estimates, and where $\hat{V}$ is an estimate of the b covariance. In the standard regression case, $\hat{V}$ is given by:

$$\hat{V}(b) = s^2\Big(\frac{\partial f(\beta)}{\partial \beta}\frac{\partial f(\beta)}{\partial \beta'}\Big)^{-1}\Big|_{\beta = b} \tag{19.7}$$

where u is the vector of unrestricted residuals, and s^2 is the usual estimator of the unrestricted residual variance, $s^2 = (u'u)/(N-k)$, but the estimator of V may differ. For example, $\hat{V}$ may be a robust variance matrix estimator computing using White or Newey-West techniques.

More formally, under the null hypothesis H_0, the Wald statistic has an asymptotic $\chi^2(q)$ distribution, where q is the number of restrictions under H_0.

For the textbook case of a linear regression model,

$$y = X\beta + \epsilon \tag{19.8}$$

and linear restrictions:

$$H_0: \ R\beta - r = 0\,, \tag{19.9}$$

where R is a known $q \times k$ matrix, and r is a q-vector, respectively. The Wald statistic in Equation (19.6) reduces to:

$$W = (Rb - r)'(Rs^2(X'X)^{-1}R')^{-1}(Rb - r)\,, \tag{19.10}$$

which is asymptotically distributed as $\chi^2(q)$ under H_0.

If we further assume that the errors ϵ are independent and identically normally distributed, we have an exact, finite sample F-statistic:

$$F = \frac{W}{q} = \frac{(\tilde{u}'\tilde{u} - u'u)/q}{(u'u)/(T-k)}, \tag{19.11}$$

where $\tilde{u}$ is the vector of residuals from the restricted regression. In this case, the F-statistic compares the residual sum of squares computed with and without the restrictions imposed.

We remind you that the expression for the finite sample F-statistic in (19.11) is for standard linear regression, and is not valid for more general cases (nonlinear models, ARMA specifications, or equations where the variances are estimated using other methods such as Newey-West or White). In non-standard settings, the reported F-statistic (which EViews always computes as W/q), does not possess the desired finite-sample properties. In these cases, while asymptotically valid, the F-statistic results should be viewed as illustrative and for comparison purposes only.

Omitted Variables

This test enables you to add a set of variables to an existing equation and to ask whether the set makes a significant contribution to explaining the variation in the dependent variable. The null hypothesis H_0 is that the additional set of regressors are not jointly significant.

The output from the test is an F-statistic and a likelihood ratio (LR) statistic with associated p-values, together with the estimation results of the unrestricted model under the alternative. The F-statistic is based on the difference between the residual sums of squares of the restricted and unrestricted regressions and is only valid in linear regression based settings. The LR statistic is computed as:

$$LR = -2(l_r - l_u) \tag{19.12}$$

where l_r and l_u are the maximized values of the (Gaussian) log likelihood function of the unrestricted and restricted regressions, respectively. Under H_0, the LR statistic has an asymptotic χ^2 distribution with degrees of freedom equal to the number of restrictions (the number of added variables).

Bear in mind that:

- The omitted variables test requires that the same number of observations exist in the original and test equations. If any of the series to be added contain missing observations over the sample of the original equation (which will often be the case when you add lagged variables), the test statistics cannot be constructed.
- The omitted variables test can be applied to equations estimated with linear LS, TSLS, ARCH (mean equation only), binary, ordered, censored, truncated, and count models. The test is available only if you specify the equation by listing the regressors, not by a formula.

To perform an LR test in these settings, you can estimate a separate equation for the unrestricted and restricted models over a common sample, and evaluate the LR statistic and p-value using scalars and the `@cchisq` function, as described above.

How to Perform an Omitted Variables Test

To test for omitted variables, select **View/Coefficient Tests/Omitted Variables-Likelihood Ratio…** In the dialog that opens, list the names of the test variables, each separated by at least one space. Suppose, for example, that the initial regression is:

```
ls log(q) c log(l) log(k)
```

If you enter the list:

```
log(m) log(e)
```

in the dialog, then EViews reports the results of the unrestricted regression containing the two additional explanatory variables, and displays statistics testing the hypothesis that the coefficients on the new variables are jointly zero. The top part of the output depicts the test results:

Omitted Variables: LOG(M) LOG(E)			
F-statistic	4.267478	Probability	0.028611
Log likelihood ratio	8.884940	Probability	0.011767

The F-statistic has an exact finite sample F-distribution under H_0 for linear models if the errors are independent and identically distributed normal random variables. The numerator degrees of freedom is the number of additional regressors and the denominator degrees of freedom is the number of observations less the total number of regressors. The log like-

lihood ratio statistic is the LR test statistic and is asymptotically distributed as a χ^2 with degrees of freedom equal to the number of added regressors.

In our example, the tests reject the null hypothesis that the two series do not belong to the equation at a 5% significance level, but cannot reject the hypothesis at a 1% significance level.

Redundant Variables

The redundant variables test allows you to test for the statistical significance of a subset of your included variables. More formally, the test is for whether a subset of variables in an equation all have zero coefficients and might thus be deleted from the equation. The redundant variables test can be applied to equations estimated by linear LS, TSLS, ARCH (mean equation only), binary, ordered, censored, truncated, and count methods. The test is available only if you specify the equation by listing the regressors, not by a formula.

How to Perform a Redundant Variables Test

To test for redundant variables, select **View/Coefficient Tests/Redundant Variables-Likelihood Ratio…** In the dialog that appears, list the names of each of the test variables, separated by at least one space. Suppose, for example, that the initial regression is:

```
ls log(q) c log(l) log(k) log(m) log(e)
```

If you type the list:

```
log(m) log(e)
```

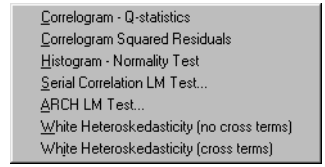
in the dialog, then EViews reports the results of the restricted regression dropping the two regressors, followed by the statistics associated with the test of the hypothesis that the coefficients on the two variables are jointly zero.

The test statistics are the F-statistic and the Log likelihood ratio. The F-statistic has an exact finite sample F-distribution under H_0 if the errors are independent and identically distributed normal random variables and the model is linear. The numerator degrees of freedom are given by the number of coefficient restrictions in the null hypothesis. The denominator degrees of freedom are given by the total regression degrees of freedom. The LR test is an asymptotic test, distributed as a χ^2 with degrees of freedom equal to the number of excluded variables under H_0. In this case, there are two degrees of freedom.

Residual Tests

EViews provides tests for serial correlation, normality, heteroskedasticity, and autoregressive conditional heteroskedasticity in the residuals from your estimated equation. Not all of these tests are available for every specification.

Correlograms and Q-statistics



This view displays the autocorrelations and partial autocorrelations of the equation residuals up to the specified number of lags. Further details on these statistics and the Ljung-Box Q-statistics that are also computed are provided in Chapter 11, "Q-Statistics" on page 328.

This view is available for the residuals from least squares, two-stage least squares, nonlinear least squares and binary, ordered, censored, and count models. In calculating the probability values for the Q-statistics, the degrees of freedom are adjusted to account for estimated ARMA terms.

To display the correlograms and Q-statistics, push **View/Residual Tests/Correlogram-Q-statistics** on the equation toolbar. In the Lag Specification dialog box, specify the number of lags you wish to use in computing the correlogram.

Correlograms of Squared Residuals

This view displays the autocorrelations and partial autocorrelations of the squared residuals up to any specified number of lags and computes the Ljung-Box Q-statistics for the corresponding lags. The correlograms of the squared residuals can be used to check autoregressive conditional heteroskedasticity (ARCH) in the residuals; see also "ARCH LM Test" on page 582, below.

If there is no ARCH in the residuals, the autocorrelations and partial autocorrelations should be zero at all lags and the Q-statistics should not be significant; see Chapter 11, page 326, for a discussion of the correlograms and Q-statistics.

This view is available for equations estimated by least squares, two-stage least squares, and nonlinear least squares estimation. In calculating the probability for Q-statistics, the degrees of freedom are adjusted for the inclusion of ARMA terms.

To display the correlograms and Q-statistics of the squared residuals, push **View/Residual Tests/Correlogram Squared Residuals** on the equation toolbar. In the Lag Specification dialog box that opens, specify the number of lags over which to compute the correlograms.

Histogram and Normality Test

This view displays a histogram and descriptive statistics of the residuals, including the Jarque-Bera statistic for testing normality. If the residuals are normally distributed, the histogram should be bell-shaped and the Jarque-Bera statistic should not be significant; see Chapter 11, page 312, for a discussion of the Jarque-Bera test. This view is available for residuals from least squares, two-stage least squares, nonlinear least squares, and binary, ordered, censored, and count models.

To display the histogram and Jarque-Bera statistic, select **View/Residual Tests/Histogram-Normality**. The Jarque-Bera statistic has a χ^2 distribution with two degrees of freedom under the null hypothesis of normally distributed errors.

Serial Correlation LM Test

This test is an alternative to the Q-statistics for testing serial correlation. The test belongs to the class of asymptotic (large sample) tests known as Lagrange multiplier (LM) tests.

Unlike the Durbin-Watson statistic for AR(1) errors, the LM test may be used to test for higher order ARMA errors and is applicable whether or not there are lagged dependent variables. Therefore, we recommend its use (in preference to the DW statistic) whenever you are concerned with the possibility that your errors exhibit autocorrelation.

The null hypothesis of the LM test is that there is no serial correlation up to lag order p, where p is a pre-specified integer. The local alternative is ARMA(r, q) errors, where the number of lag terms $p = \max(r, q)$. Note that this alternative includes both AR(p) and MA(p) error processes, so that the test may have power against a variety of alternative autocorrelation structures. See Godfrey (1988), for further discussion.

The test statistic is computed by an auxiliary regression as follows. First, suppose you have estimated the regression;

$$y_t = X_t\beta + \epsilon_t \tag{19.13}$$

where b are the estimated coefficients and ϵ are the errors. The test statistic for lag order p is based on the auxiliary regression for the residuals $e = y - X\hat{\beta}$:

$$e_t = X_t\gamma + \left(\sum_{s=1}^{p} \alpha_s e_{t-s} \right) + v_t . \tag{19.14}$$

Following the suggestion by Davidson and MacKinnon (1993), EViews sets any presample values of the residuals to 0. This approach does not affect the asymptotic distribution of the statistic, and Davidson and MacKinnon argue that doing so provides a test statistic which has better finite sample properties than an approach which drops the initial observations.

This is a regression of the residuals on the original regressors X and lagged residuals up to order p. EViews reports two test statistics from this test regression. The F-statistic is an omitted variable test for the joint significance of all lagged residuals. Because the omitted variables are residuals and not independent variables, the exact finite sample distribution of the F-statistic under H_0 is still not known, but we present the F-statistic for comparison purposes.

The Obs*R-squared statistic is the Breusch-Godfrey LM test statistic. This LM statistic is computed as the number of observations, times the (uncentered) R^2 from the test regres-

sion. Under quite general conditions, the LM test statistic is asymptotically distributed as a $\chi^2(p)$.

The serial correlation LM test is available for residuals from either least squares or two-stage least squares estimation. The original regression may include AR and MA terms, in which case the test regression will be modified to take account of the ARMA terms. Testing in 2SLS settings involves additional complications, see Wooldridge (1990) for details.

To carry out the test, push **View/Residual Tests/Serial Correlation LM Test…** on the equation toolbar and specify the highest order of the AR or MA process that might describe the serial correlation. If the test indicates serial correlation in the residuals, LS standard errors are invalid and should not be used for inference.

ARCH LM Test

This is a Lagrange multiplier (LM) test for autoregressive conditional heteroskedasticity (ARCH) in the residuals (Engle 1982). This particular specification of heteroskedasticity was motivated by the observation that in many financial time series, the magnitude of residuals appeared to be related to the magnitude of recent residuals. ARCH in itself does not invalidate standard LS inference. However, ignoring ARCH effects may result in loss of efficiency; see Chapter 20 for a discussion of estimation of ARCH models in EViews.

The ARCH LM test statistic is computed from an auxiliary test regression. To test the null hypothesis that there is no ARCH up to order q in the residuals, we run the regression:

$$e_t^2 = \beta_0 + \left(\sum_{s=1}^{q} \beta_s e_{t-s}^2 \right) + v_t , \tag{19.15}$$

where e is the residual. This is a regression of the squared residuals on a constant and lagged squared residuals up to order q. EViews reports two test statistics from this test regression. The F-statistic is an omitted variable test for the joint significance of all lagged squared residuals. The Obs*R-squared statistic is Engle's LM test statistic, computed as the number of observations times the R^2 from the test regression. The exact finite sample distribution of the F-statistic under H_0 is not known but the LM test statistic is asymptotically distributed $\chi^2(q)$ under quite general conditions. The ARCH LM test is available for equations estimated by least squares, two-stage least squares, and nonlinear least squares.

To carry out the test, push **View/Residual Tests/ARCH LM Test…** on the equation toolbar and specify the order of ARCH to be tested against.

White's Heteroskedasticity Test

This is a test for heteroskedasticity in the residuals from a least squares regression (White, 1980). Ordinary least squares estimates are consistent in the presence of heteroskedasticity, but the conventional computed standard errors are no longer valid. If you find evi-

dence of heteroskedasticity, you should either choose the robust standard errors option to correct the standard errors (see "Heteroskedasticity Consistent Covariances (White)" on page 472) or you should model the heteroskedasticity to obtain more efficient estimates using weighted least squares.

White's test is a test of the null hypothesis of no heteroskedasticity against heteroskedasticity of some unknown general form. The test statistic is computed by an auxiliary regression, where we regress the squared residuals on all possible (nonredundant) cross products of the regressors. For example, suppose we estimated the following regression:

$$y_t = b_1 + b_2 x_t + b_3 z_t + e_t \tag{19.16}$$

where the b are the estimated parameters and e the residual. The test statistic is then based on the auxiliary regression:

$$e_t^2 = \alpha_0 + \alpha_1 x_t + \alpha_2 z_t + \alpha_3 x_t^2 + \alpha_4 z_t^2 + \alpha_5 x_t z_t + v_t \,. \tag{19.17}$$

EViews reports two test statistics from the test regression. The F-statistic is an omitted variable test for the joint significance of all cross products, excluding the constant. It is presented for comparison purposes.

The Obs*R-squared statistic is White's test statistic, computed as the number of observations times the centered R^2 from the test regression. The exact finite sample distribution of the F-statistic under H_0 is not known, but White's test statistic is asymptotically distributed as a χ^2 with degrees of freedom equal to the number of slope coefficients (excluding the constant) in the test regression.

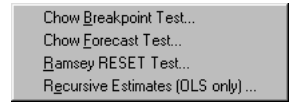
White also describes this approach as a general test for model misspecification, since the null hypothesis underlying the test assumes that the errors are both homoskedastic and independent of the regressors, and that the linear specification of the model is correct. Failure of any one of these conditions could lead to a significant test statistic. Conversely, a non-significant test statistic implies that none of the three conditions is violated.

When there are redundant cross-products, EViews automatically drops them from the test regression. For example, the square of a dummy variable is the dummy variable itself, so that EViews drops the squared term to avoid perfect collinearity.

To carry out White's heteroskedasticity test, select **View/Residual Tests/White Heteroskedasticity**. EViews has two options for the test: cross terms and no cross terms. The cross terms version of the test is the original version of White's test that includes all of the cross product terms (in the example above, $x_t z_t$). However, with many right-hand side variables in the regression, the number of possible cross product terms becomes very large so that it may not be practical to include all of them. The no cross terms option runs the test regression using only squares of the regressors.

Specification and Stability Tests

EViews provides a number of test statistic views that examine whether the parameters of your model are stable across various subsamples of your data.



One recommended empirical technique is to split the T observations in your data set of observations into T_1 observations to be used for estimation, and $T_2 = T - T_1$ observations to be used for testing and evaluation. Using all available sample observations for estimation promotes a search for a specification that best fits that specific data set, but does not allow for testing predictions of the model against data that have not been used in estimating the model. Nor does it allow one to test for parameter constancy, stability and robustness of the estimated relationship. In time series work, you will usually take the first T_1 observations for estimation and the last T_2 for testing. With cross-section data, you may wish to order the data by some variable, such as household income, sales of a firm, or other indicator variables and use a sub-set for testing.

There are no hard and fast rules for determining the relative sizes of T_1 and T_2. In some cases there may be obvious points at which a break in structure might have taken place—a war, a piece of legislation, a switch from fixed to floating exchange rates, or an oil shock. Where there is no reason *a priori* to expect a structural break, a commonly used rule-of-thumb is to use 85 to 90 percent of the observations for estimation and the remainder for testing.

EViews provides built-in procedures which facilitate variations on this type of analysis.

Chow's Breakpoint Test

The idea of the breakpoint Chow test is to fit the equation separately for each subsample and to see whether there are significant differences in the estimated equations. A significant difference indicates a structural change in the relationship. For example, you can use this test to examine whether the demand function for energy was the same before and after the oil shock. The test may be used with least squares and two-stage least squares regressions.

To carry out the test, we partition the data into two or more subsamples. Each subsample must contain more observations than the number of coefficients in the equation so that the equation can be estimated. The Chow breakpoint test compares the sum of squared residuals obtained by fitting a single equation to the entire sample with the sum of squared residuals obtained when separate equations are fit to each subsample of the data.

EViews reports two test statistics for the Chow breakpoint test. The F-statistic is based on the comparison of the restricted and unrestricted sum of squared residuals and in the simplest case involving a single breakpoint, is computed as:

$$F = \frac{(\tilde{u}'\tilde{u} - (u_1'u_1 + u_2'u_2))/k}{(u_1'u_1 + u_2'u_2)/(T-2k)}, \tag{19.18}$$

where $\tilde{u}'\tilde{u}$ is the restricted sum of squared residuals, $u_i'u_i$ is the sum of squared residuals from subsample i, T is the total number of observations, and k is the number of parameters in the equation. This formula can be generalized naturally to more than one breakpoint. The *F*-statistic has an exact finite sample *F*-distribution if the errors are independent and identically distributed normal random variables.

The log likelihood ratio statistic is based on the comparison of the restricted and unrestricted maximum of the (Gaussian) log likelihood function. The LR test statistic has an asymptotic χ^2 distribution with degrees of freedom equal to $(m-1)k$ under the null hypothesis of no structural change, where m is the number of subsamples.

One major drawback of the breakpoint test is that each subsample requires at least as many observations as the number of estimated parameters. This may be a problem if, for example, you want to test for structural change between wartime and peacetime where there are only a few observations in the wartime sample. The Chow forecast test, discussed below, should be used in such cases.

To apply the Chow breakpoint test, push **View/Stability Tests/Chow Breakpoint Test...** on the equation toolbar. In the dialog that appears, list the dates or observation numbers for the breakpoints. For example, if your original equation was estimated from 1950 to 1994, entering:

```
1960
```

in the dialog specifies two subsamples, one from 1950 to 1959 and one from 1960 to 1994. Typing:

```
1960 1970
```

specifies three subsamples, 1950 to 1959, 1960 to 1969, and 1970 to 1994.

Chow's Forecast Test

The Chow forecast test estimates two models—one using the full set of data T, and the other using a long subperiod T_1. A long difference between the two models casts doubt on the stability of the estimated relation over the sample period. The Chow forecast test can be used with least squares and two-stage least squares regressions.

EViews reports two test statistics for the Chow forecast test. The *F*-statistic is computed as

$$F = \frac{(\tilde{u}'\tilde{u} - u'u)/T_2}{u'u/(T_1-k)}, \tag{19.19}$$

where $\tilde{u}'\tilde{u}$ is the residual sum of squares when the equation is fitted to all T sample observations, $u'u$ is the residual sum of squares when the equation is fitted to T_1 observations, and k is the number of estimated coefficients. This F-statistic follows an exact finite sample F-distribution if the errors are independent, and identically, normally distributed.

The log likelihood ratio statistic is based on the comparison of the restricted and unrestricted maximum of the (Gaussian) log likelihood function. Both the restricted and unrestricted log likelihood are obtained by estimating the regression using the whole sample. The restricted regression uses the original set of regressors, while the unrestricted regression adds a dummy variable for each forecast point. The LR test statistic has an asymptotic χ^2 distribution with degrees of freedom equal to the number of forecast points T_2 under the null hypothesis of no structural change.

To apply Chow's forecast test, push **View/Stability Tests/Chow Forecast Test...** on the equation toolbar and specify the date or observation number for the beginning of the forecasting sample. The date should be within the current sample of observations.

As an example, suppose we estimate a consumption function using quarterly data from 1947:1 to 1994:4 and specify 1973:1 as the first observation in the forecast period. The test reestimates the equation for the period 1947:1 to 1972:4, and uses the result to compute the prediction errors for the remaining quarters, and reports the following results:

Chow Forecast Test: Forecast from 1973:1 to 1994:4			
F-statistic	0.708348	Probability	0.951073
Log likelihood ratio	91.57088	Probability	0.376108

Neither of the forecast test statistics reject the null hypothesis of no structural change in the consumption function before and after 1973:1.

If we test the same hypothesis using the Chow breakpoint test, the result is:

Chow Breakpoint Test: 1973:1			
F-statistic	38.39198	Probability	0.000000
Log likelihood ratio	65.75468	Probability	0.000000

Note that both of the breakpoint test statistics decisively reject the hypothesis from above. This example illustrates the possibility that the two Chow tests may yield conflicting results.

Ramsey's RESET Test

RESET stands for *Regression Specification Error Test* and was proposed by Ramsey (1969). The classical normal linear regression model is specified as:

$$y = X\beta + \epsilon, \tag{19.20}$$

where the disturbance vector ϵ is presumed to follow the multivariate normal distribution $N(0, \sigma^2 I)$. Specification error is an omnibus term which covers any departure from the assumptions of the maintained model. Serial correlation, heteroskedasticity, or non-normality of all violate the assumption that the disturbances are distributed $N(0, \sigma^2 I)$. Tests for these specification errors have been described above. In contrast, RESET is a general test for the following types of specification errors:

- Omitted variables; X does not include all relevant variables.
- Incorrect functional form; some or all of the variables in y and X should be transformed to logs, powers, reciprocals, or in some other way.
- Correlation between X and ϵ, which may be caused, among other things, by measurement error in X, simultaneity, or the presence of lagged y values and serially correlated disturbances.

Under such specification errors, LS estimators will be biased and inconsistent, and conventional inference procedures will be invalidated. Ramsey (1969) showed that any or all of these specification errors produce a non-zero mean vector for ϵ. Therefore, the null and alternative hypotheses of the RESET test are:

$$\begin{aligned} H_0&: \epsilon \sim N(0, \sigma^2 I) \\ H_1&: \epsilon \sim N(\mu, \sigma^2 I) \qquad \mu \neq 0 \end{aligned} \tag{19.21}$$

The test is based on an augmented regression:

$$y = X\beta + Z\gamma + \epsilon. \tag{19.22}$$

The test of specification error evaluates the restriction $\gamma = 0$. The crucial question in constructing the test is to determine what variables should enter the Z matrix. Note that the Z matrix may, for example, be comprised of variables that are not in the original specification, so that the test of $\gamma = 0$ is simply the omitted variables test described above.

In testing for incorrect functional form, the nonlinear part of the regression model may be some function of the regressors included in X. For example, if a linear relation,

$$y = \beta_0 + \beta_1 X + \epsilon, \tag{19.23}$$

is specified instead of the true relation:

$$y = \beta_0 + \beta_1 X + \beta_2 X^2 + \epsilon \tag{19.24}$$

the augmented model has $Z = X^2$ and we are back to the omitted variable case. A more general example might be the specification of an additive relation,

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon \tag{19.25}$$

instead of the (true) multiplicative relation:

$$y = \beta_0 X_1^{\beta_1} X_2^{\beta_2} + \epsilon \,. \tag{19.26}$$

A Taylor series approximation of the multiplicative relation would yield an expression involving powers and cross-products of the explanatory variables. Ramsey's suggestion is to include powers of the predicted values of the dependent variable (which are, of course, linear combinations of powers and cross-product terms of the explanatory variables) in Z:

$$Z = [\hat{y}^2, \hat{y}^3, \hat{y}^4, \ldots] \tag{19.27}$$

where $\hat{y}$ is the vector of fitted values from the regression of y on X. The superscripts indicate the powers to which these predictions are raised. The first power is not included since it is perfectly collinear with the X matrix.

Output from the test reports the test regression and the F-statistic and log likelihood ratio for testing the hypothesis that the coefficients on the powers of fitted values are all zero. A study by Ramsey and Alexander (1984) showed that the RESET test could detect specification error in an equation which was known *a priori* to be misspecified but which nonetheless gave satisfactory values for all the more traditional test criteria—goodness of fit, test for first order serial correlation, high t-ratios.

To apply the test, select **View/Stability Tests/Ramsey RESET Test...** and specify the number of fitted terms to include in the test regression. The fitted terms are the powers of the fitted values from the original regression, starting with the square or second power. For example, if you specify 1, then the test will add $\hat{y}^2$ in the regression, and if you specify 2, then the test will add $\hat{y}^2$ and $\hat{y}^3$ in the regression, and so on. If you specify a large number of fitted terms, EViews may report a near singular matrix error message since the powers of the fitted values are likely to be highly collinear. The Ramsey RESET test is applicable only to an equation estimated by least squares.

Recursive Least Squares

In recursive least squares the equation is estimated repeatedly, using ever larger subsets of the sample data. If there are k coefficients to be estimated in the b vector, then the first k observations are used to form the first estimate of b. The next observation is then added to the data set and $k+1$ observations are used to compute the second estimate of b. This process is repeated until all the T sample points have been used, yielding $T-k+1$ estimates of the b vector. At each step the last estimate of b can be used to predict the next value of the dependent variable. The one-step ahead forecast error resulting from this prediction, suitably scaled, is defined to be a *recursive residual.*

More formally, let X_{t-1} denote the $(t-1)\times k$ matrix of the regressors from period 1 to period $t-1$, and y_{t-1} the corresponding vector of observations on the dependent vari-

able. These data up to period $t-1$ give an estimated coefficient vector, denoted by b_{t-1}. This coefficient vector gives you a forecast of the dependent variable in period t. The forecast is $x_t'b$, where x_t is the row vector of observations on the regressors in period t. The forecast error is $y_t - x_t'b$, and the forecast variance is given by:

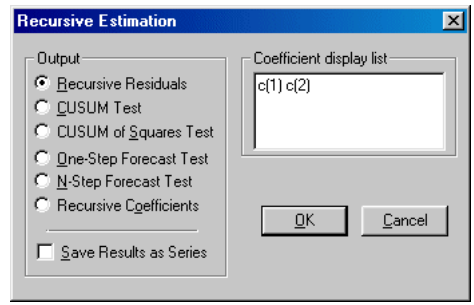
$$\sigma^2(1 + x_t'(X_t'X_t)^{-1}x_t)\,. \tag{19.28}$$

The recursive residual w_t is defined in EViews as:

$$w_t = \frac{(y_t - x_t'b)}{(1 + x_t'(X_t'X_t)^{-1}x_t)^{1/2}}\,. \tag{19.29}$$

These residuals can be computed for $t = k+1, \ldots, T$. If the maintained model is valid, the recursive residuals will be independently and normally distributed with zero mean and constant variance σ^2.

To calculate the recursive residuals, press **View/Stability Tests/Recursive Estimates (OLS only)…** on the equation toolbar. There are six options available for the recursive estimates view. The recursive estimates view is only available for equations estimated by ordinary least squares without AR and MA terms. The **Save Results as Series** option allows you to save the recursive residuals and recursive coefficients as named series in the workfile; see "Save Results as Series" on page 592.



Recursive Residuals

This option shows a plot of the recursive residuals about the zero line. Plus and minus two standard errors are also shown at each point. Residuals outside the standard error bands suggest instability in the parameters of the equation.

CUSUM Test

The CUSUM test (Brown, Durbin, and Evans, 1975) is based on the cumulative sum of the recursive residuals. This option plots the cumulative sum together with the 5% critical lines. The test finds parameter instability if the cumulative sum goes outside the area between the two critical lines.

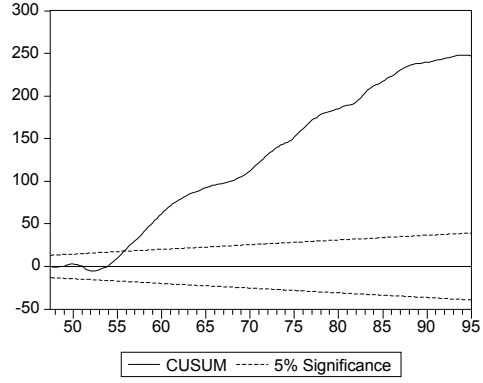
The CUSUM test is based on the statistic:

$$W_t = \sum_{r=k+1}^{t} w_r / s\,, \tag{19.30}$$

for $t = k+1, \ldots, T$, where w is the recursive residual defined above, and s is the standard error of the regression fitted to all T sample points. If the β vector remains constant from period to period, $\mathrm{E}(W_t) = 0$, but if β changes, W_t will tend to diverge from the zero mean value line. The significance of any departure from the zero line is assessed by reference to a pair of 5% significance lines, the distance between which increases with t. The 5% significance lines are found by connecting the points:

$$[k, \pm -0.948(T-k)^{1/2}] \qquad \text{and} \qquad [T, \pm 3 \times 0.948(T-k)^{1/2}]. \tag{19.31}$$

Movement of W_t outside the critical lines is suggestive of coefficient instability. A sample CUSUM test is given below:



The test clearly indicates instability in the equation during the sample period.

CUSUM of Squares Test

The CUSUM of squares test (Brown, Durbin, and Evans, 1975) is based on the test statistic:

$$W_t = \left(\sum_{r=k+1}^{t} w_r^2 \right) / \left(\sum_{r=k+1}^{T} w_r^2 \right). \tag{19.32}$$

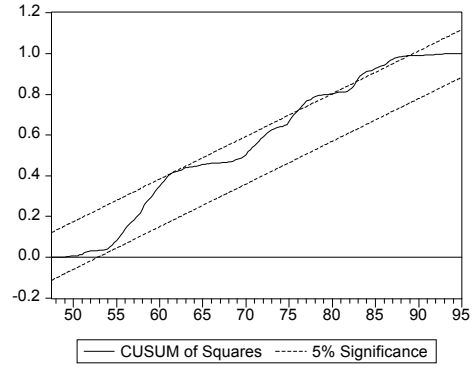
The expected value of S under the hypothesis of parameter constancy is:

$$\mathrm{E}(S_t) = (t-k)/(T-k) \tag{19.33}$$

which goes from zero at $t = k$ to unity at $t = T$. The significance of the departure of S from its expected value is assessed by reference to a pair of parallel straight lines around the expected value. See Brown, Durbin, and Evans (1975) or Johnston and DiNardo (1997, Table D.8) for a table of significance lines for the CUSUM of squares test.

The CUSUM of squares test provides a plot of S_t against t and the pair of 5 percent critical lines. As with the CUSUM test, movement outside the critical lines is suggestive of parameter or variance instability.

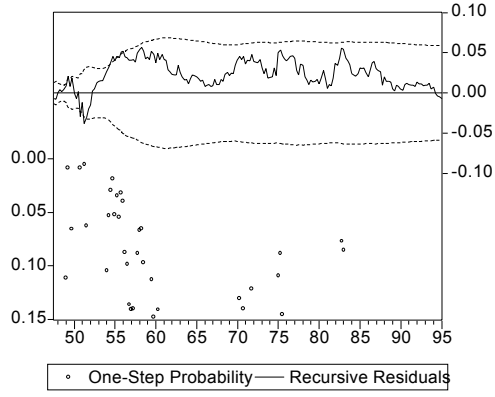
The cumulative sum of squares is generally within the 5% significance lines, suggesting that the residual variance is somewhat stable.



One-Step Forecast Test

If you look back at the definition of the recursive residuals given above, you will see that each recursive residual is the error in a one-step ahead forecast. To test whether the value of the dependent variable at time t might have come from the model fitted to all the data up to that point, each error can be compared with its standard deviation from the full sample.

The **One-Step Forecast Test** option produces a plot of the recursive residuals and standard errors and the sample points whose probability value is at or below 15 percent. The plot can help you spot the periods when your equation is least successful. For example, the one-step ahead forecast test might look like this:



The upper portion of the plot (right vertical axis) repeats the recursive residuals and standard errors displayed by the **Recursive Residuals** option. The lower portion of the plot (left vertical axis) shows the probability values for those sample points where the hypothesis of parameter constancy would be rejected at the 5, 10, or 15 percent levels. The points with p-values less the 0.05 correspond to those points where the recursive residuals go outside the two standard error bounds.

For the test equation, there is evidence of instability early in the sample period.

N-Step Forecast Test

This test uses the recursive calculations to carry out a sequence of Chow Forecast tests. In contrast to the single Chow Forecast test described earlier, this test does not require the specification of a forecast period— it automatically computes all feasible cases, starting with the smallest possible sample size for estimating the forecasting equation and then adding one observation at a time. The plot from this test shows the recursive residuals at

the top and significant probabilities (based on the *F*-statistic) in the lower portion of the diagram.

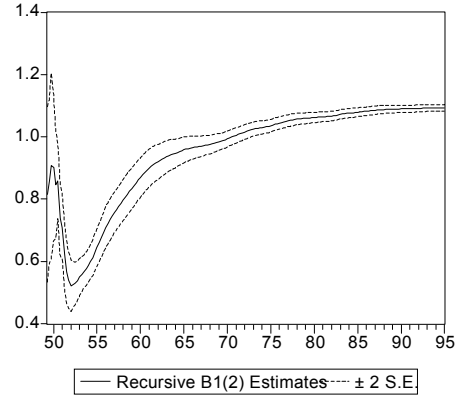
Recursive Coefficient Estimates

This view enables you to trace the evolution of estimates for any coefficient as more and more of the sample data are used in the estimation. The view will provide a plot of selected coefficients in the equation for all feasible recursive estimations. Also shown are the two standard error bands around the estimated coefficients.

If the coefficient displays significant variation as more data is added to the estimating equation, it is a strong indication of instability. Coefficient plots will sometimes show dramatic jumps as the postulated equation tries to digest a structural break.

To view the recursive coefficient estimates, click the **Recursive Coefficients** option and list the coefficients you want to plot in the **Coefficient Display List** field of the dialog box. The recursive estimates of the marginal propensity to consume (coefficient C(2)), from the sample consumption function are provided below:

The estimated propensity to consume rises steadily as we add more data over the sample period, approaching a value of one.



Save Results as Series

The **Save Results as Series** checkbox will do different things depending on the plot you have asked to be displayed. When paired with the **Recursive Coefficients** option, **Save Results as Series** will instruct EViews to save all recursive coefficients and their standard errors in the workfile as named series. EViews will name the coefficients using the next available name of the form, R_C1, R_C2, …, and the corresponding standard errors as R_C1SE, R_C2SE, and so on.

If you check the **Save Results as Series** box with any of the other options, EViews saves the recursive residuals and the recursive standard errors as named series in the workfile. EViews will name the residual and standard errors as R_RES and R_RESSE, respectively.

Note that you can use the recursive residuals to reconstruct the CUSUM and CUSUM of squares series.

Applications

For illustrative purposes, we provide a demonstration of how to carry out some other specification tests in EViews. For brevity, the discussion is based on commands, but most of these procedures can also be carried out using the menu system.

A Wald test of structural change with unequal variance

The F-statistics reported in the Chow tests have an F-distribution only if the errors are independent and identically normally distributed. This restriction implies that the residual variance in the two subsamples must be equal.

Suppose now that we wish to compute a Wald statistic for structural change with unequal subsample variances. Denote the parameter estimates and their covariance matrix in subsample i as b_i and V_i for $i = 1, 2$. Under the assumption that b_1 and b_2 are independent normal random variables, the difference $b_1 - b_2$ has mean zero and variance $V_1 + V_2$. Therefore, a Wald statistic for the null hypothesis of no structural change and independent samples can be constructed as:

$$W = (b_1 - b_2)'(V_1 + V_2)^{-1}(b_1 - b_2), \tag{19.34}$$

which has an asymptotic χ^2 distribution with degrees of freedom equal to the number of estimated parameters in the b vector.

To carry out this test in EViews, we estimate the model in each subsample and save the estimated coefficients and their covariance matrix. For example, suppose we have a quarterly sample of 1947:1–1994:4 and wish to test whether there was a structural change in the consumption function in 1973:1. First, estimate the model in the first sample and save the results by the commands:

```
coef(2) b1
smpl 1947:1 1972:4
equation eq_1.ls log(cs)=b1(1)+b1(2)*log(gdp)
sym v1=eq_1.@cov
```

The first line declares the coefficient vector, B1, into which we will place the coefficient estimates in the first sample. Note that the equation specification in the third line explicitly refers to elements of this coefficient vector. The last line saves the coefficient covariance matrix as a symmetric matrix named V1. Similarly, estimate the model in the second sample and save the results by the commands:

```
coef(2) b2
smpl 1973:1 1994:4
equation eq_2.ls log(cs)=b2(1)+b2(2)*log(gdp)
sym v2=eq_2.@cov
```

To compute the Wald statistic, use the command:

```
matrix wald=@transpose(b1-b2)*@inverse(v1+v2)*(b1-b2)
```

The Wald statistic is saved in the 1×1 matrix named WALD. To see the value, either double click on WALD or type "show wald". You can compare this value with the critical values from the χ^2 distribution with 2 degrees of freedom. Alternatively, you can compute the p-value in EViews using the command:

```
scalar wald_p=1-@cchisq(wald(1,1),2)
```

The p-value is saved as a scalar named WALD_P. To see the p-value, double click on WALD_P or type "show wald_p". The p-value will be displayed in the status line at the bottom of the EViews window.

The Hausman test

A widely used class of tests in econometrics is the Hausman test. The underlying idea of the Hausman test is to compare two sets of estimates, one of which is consistent under both the null and the alternative and another which is consistent only under the null hypothesis. A large difference between the two sets of estimates is taken as evidence in favor of the alternative hypothesis.

Hausman (1978) originally proposed a test statistic for endogeneity based upon a direct comparison of coefficient values. Here, we illustrate the version of the Hausman test proposed by Davidson and MacKinnon (1989, 1993), which carries out the test by running an auxiliary regression.

The following equation was estimated by OLS:

Dependent Variable: LOG(M1)
Method: Least Squares
Date: 08/13/97 Time: 14:12
Sample(adjusted): 1959:02 1995:04
Included observations: 435 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.022699	0.004443	-5.108528	0.0000
LOG(IP)	0.011630	0.002585	4.499708	0.0000
DLOG(PPI)	-0.024886	0.042754	-0.582071	0.5608
TB3	-0.000366	9.91E-05	-3.692675	0.0003
LOG(M1(-1))	0.996578	0.001210	823.4440	0.0000

R-squared	0.999953	Mean dependent var	5.844581
Adjusted R-squared	0.999953	S.D. dependent var	0.670596
S.E. of regression	0.004601	Akaike info criterion	-7.913714
Sum squared resid	0.009102	Schwarz criterion	-7.866871
Log likelihood	1726.233	F-statistic	2304897.
Durbin-Watson stat	1.265920	Prob(F-statistic)	0.000000

Suppose we are concerned that industrial production (IP) is endogenously determined with money (M1) through the money supply function. If endogeneity is present, then OLS estimates will be biased and inconsistent. To test this hypothesis, we need to find a set of instrumental variables that are correlated with the "suspect" variable IP but not with the error term of the money demand equation. The choice of the appropriate instrument is a crucial step. Here, we take the unemployment rate (URATE) and Moody's AAA corporate bond yield (AAA) as instruments.

To carry out the Hausman test by artificial regression, we run two OLS regressions. In the first regression, we regress the suspect variable (log) IP on all exogenous variables and instruments and retrieve the residuals:

```
ls log(ip) c dlog(ppi) tb3 log(m1(-1)) urate aaa
series res_ip=resid
```

Then in the second regression, we re-estimate the money demand function including the residuals from the first regression as additional regressors. The result is:

Dependent Variable: LOG(M1)
Method: Least Squares
Date: 08/13/97 Time: 15:28
Sample(adjusted): 1959:02 1995:04
Included observations: 435 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.007145	0.007473	-0.956158	0.3395
LOG(IP)	0.001560	0.004672	0.333832	0.7387
DLOG(PPI)	0.020233	0.045935	0.440465	0.6598
TB3	-0.000185	0.000121	-1.527775	0.1273
LOG(M1(-1))	1.001093	0.002123	471.4894	0.0000
RES_IP	0.014428	0.005593	2.579826	0.0102

R-squared	0.999954	Mean dependent var	5.844581
Adjusted R-squared	0.999954	S.D. dependent var	0.670596
S.E. of regression	0.004571	Akaike info criterion	-7.924512
Sum squared resid	0.008963	Schwarz criterion	-7.868300
Log likelihood	1729.581	F-statistic	1868171.
Durbin-Watson stat	1.307838	Prob(F-statistic)	0.000000

If the OLS estimates are consistent, then the coefficient on the first stage residuals should not be significantly different from zero. In this example, the test (marginally) rejects the hypothesis of consistent OLS estimates (to be more precise, this is an asymptotic test and you should compare the *t*-statistic with the critical values from the standard normal).

Non-nested Tests

Most of the tests discussed in this chapter are nested tests in which the null hypothesis is obtained as a special case of the alternative hypothesis. Now consider the problem of choosing between the following two specifications of a consumption function:

$$\begin{aligned} H_1: \quad & CS_t = \alpha_1 + \alpha_2 GDP_t + \alpha_3 GDP_{t-1} + \epsilon_t \\ H_2: \quad & CS_t = \beta_1 + \beta_2 GDP_t + \beta_3 CS_{t-1} + \epsilon_t \end{aligned} \tag{19.35}$$

These are examples of non-nested models since neither model may be expressed as a restricted version of the other.

The J-test proposed by Davidson and MacKinnon (1993) provides one method of choosing between two non-nested models. The idea is that if one model is the correct model, then the fitted values from the other model should not have explanatory power when estimating that model. For example, to test model H_1 against model H_2, we first estimate model H_2 and retrieve the fitted values:

```
equation eq_cs2.ls cs c gdp cs(-1)
eq_cs2.fit cs2
```

The second line saves the fitted values as a series named CS2. Then estimate model H_1 including the fitted values from model H_2. The result is:

Dependent Variable: CS
Method: Least Squares
Date: 8/13/97 Time: 00:49
Sample(adjusted): 1947:2 1994:4
Included observations: 191 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	7.313232	4.391305	1.665389	0.0975
GDP	0.278749	0.029278	9.520694	0.0000
GDP(-1)	-0.314540	0.029287	-10.73978	0.0000
CS2	1.048470	0.019684	53.26506	0.0000

R-squared	0.999833	Mean dependent var	1953.966
Adjusted R-squared	0.999830	S.D. dependent var	848.4387
S.E. of regression	11.05357	Akaike info criterion	7.664104
Sum squared resid	22847.93	Schwarz criterion	7.732215
Log likelihood	-727.9219	F-statistic	373074.4
Durbin-Watson stat	2.253186	Prob(F-statistic)	0.000000

The fitted values from model H_2 enter significantly in model H_1 and we reject model H_1.

We must also test model H_2 against model H_1. Estimate model H_1, retrieve the fitted values, and estimate model H_2 including the fitted values from model H_1. The results of this "reverse" test are given by:

Dependent Variable: CS
Method: Least Squares
Date: 08/13/97 Time: 16:58
Sample(adjusted): 1947:2 1994:4
Included observations: 191 after adjusting endpoints

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-1427.716	132.0349	-10.81318	0.0000
GDP	5.170543	0.476803	10.84419	0.0000
CS(-1)	0.977296	0.018348	53.26506	0.0000
CS1	-7.292771	0.679043	-10.73978	0.0000

R-squared	0.999833	Mean dependent var	1953.966
Adjusted R-squared	0.999830	S.D. dependent var	848.4387
S.E. of regression	11.05357	Akaike info criterion	7.664104
Sum squared resid	22847.93	Schwarz criterion	7.732215
Log likelihood	-727.9219	F-statistic	373074.4
Durbin-Watson stat	2.253186	Prob(F-statistic)	0.000000

The fitted values are again statistically significant and we reject model H_2.

In this example, we reject both specifications, against the alternatives, suggesting that another model for the data is needed. It is also possible that we fail to reject both models, in which case the data do not provide enough information to discriminate between the two models.

Part IV. Advanced Single Equation Analysis

The following sections describe EViews tools for the estimation and analysis of advanced single equation models.

- Chapter 20, "ARCH and GARCH Estimation", beginning on page 601, outlines the EViews tools for ARCH and GARCH modeling of the conditional variance, or volatility, of a variable.
- Chapter 21, "Discrete and Limited Dependent Variable Models", on page 621 documents EViews tools for estimating qualitative and limited dependent variable models. EViews provides estimation routines for binary or ordered (probit, logit, gompit), censored or truncated (tobit, etc.), and integer valued (count data).
- Chapter 22, "The Log Likelihood (LogL) Object", beginning on page 671 describes techniques for using EViews to estimate the parameters of maximum likelihood models where you may specify the form of the likelihood.

Multiple equation models and forecasting are described in Part V. "Multiple Equation Analysis" beginning on page 693.

Chapter 29, "Panel Estimation", beginning on page 901 describes estimation in panel structured workfiles.

Chapter 20. ARCH and GARCH Estimation

Most of the statistical tools in EViews are designed to model the conditional mean of a random variable. The tools described in this chapter differ by modeling the conditional variance, or volatility, of a variable.

There are several reasons that you may wish to model and forecast volatility. First, you may need to analyze the risk of holding an asset or the value of an option. Second, forecast confidence intervals may be time-varying, so that more accurate intervals can be obtained by modeling the variance of the errors. Third, more efficient estimators can be obtained if heteroskedasticity in the errors is handled properly.

Autoregressive Conditional Heteroskedasticity (ARCH) models are specifically designed to model and forecast conditional variances. The variance of the dependent variable is modeled as a function of past values of the dependent variable and independent, or exogenous variables.

ARCH models were introduced by Engle (1982) and generalized as GARCH (Generalized ARCH) by Bollerslev (1986) and Taylor (1986). These models are widely used in various branches of econometrics, especially in financial time series analysis. See Bollerslev, Chou, and Kroner (1992) and Bollerslev, Engle, and Nelson (1994) for recent surveys.

In the next section, the basic ARCH model will be described in detail. In subsequent sections, we consider the wide range of specifications available in EViews for modeling volatility. For brevity of discussion, we will use ARCH to refer to both ARCH and GARCH models, except where there is the possibility of confusion.

Basic ARCH Specifications

In developing an ARCH model, you will have to provide three distinct specifications—one for the conditional mean equation, one for the conditional variance, and one for the conditional error distribution. We begin by describing some basic specifications for these terms. The discussion of more complicated models is taken up in "Additional ARCH Models" on page 612.

The GARCH(1, 1) Model

We begin with the simplest GARCH(1,1) specification:

$$Y_t = X_t'\theta + \epsilon_t \tag{20.1}$$

$$\sigma_t^2 = \omega + \alpha\epsilon_{t-1}^2 + \beta\sigma_{t-1}^2 \tag{20.2}$$

in which the mean equation given in (20.1) is written as a function of exogenous variables with an error term. Since σ_t^2 is the one-period ahead forecast variance based on past infor-

mation, it is called the *conditional variance*. The conditional variance equation specified in (20.2) is a function of three terms:

- A constant term: ω.
- News about volatility from the previous period, measured as the lag of the squared residual from the mean equation: ϵ_{t-1}^2 (the ARCH term).
- Last period's forecast variance: σ_{t-1}^2 (the GARCH term).

The (1, 1) in GARCH(1, 1) refers to the presence of a first-order autoregressive GARCH term (the first term in parentheses) and a first-order moving average ARCH term (the second term in parentheses). An ordinary ARCH model is a special case of a GARCH specification in which there are no lagged forecast variances in the conditional variance equation—*i.e.,* a GARCH(0, 1).

This specification is often interpreted in a financial context, where an agent or trader predicts this period's variance by forming a weighted average of a long term average (the constant), the forecasted variance from last period (the GARCH term), and information about volatility observed in the previous period (the ARCH term). If the asset return was unexpectedly large in either the upward or the downward direction, then the trader will increase the estimate of the variance for the next period. This model is also consistent with the volatility clustering often seen in financial returns data, where large changes in returns are likely to be followed by further large changes.

There are two equivalent representations of the variance equation that may aid you in interpreting the model:

- If we recursively substitute for the lagged variance on the right-hand side of Equation (20.2), we can express the conditional variance as a weighted average of all of the lagged squared residuals:

$$\sigma_t^2 = \frac{\omega}{(1-\beta)} + \alpha \sum_{j=1}^{\infty} \beta^{j-1} \epsilon_{t-j}^2 . \tag{20.3}$$

  We see that the GARCH(1,1) variance specification is analogous to the sample variance, but that it down-weights more distant lagged squared errors.

- The error in the squared returns is given by $\upsilon_t = \epsilon_t^2 - \sigma_t^2$. Substituting for the variances in the variance equation and rearranging terms we can write our model in terms of the errors:

$$\epsilon_t^2 = \omega + (\alpha + \beta)\epsilon_{t-1}^2 + \upsilon_t - \beta\nu_{t-1} . \tag{20.4}$$

  Thus, the squared errors follow a heteroskedastic ARMA(1,1) process. The autoregressive root which governs the persistence of volatility shocks is the sum of α plus

β. In many applied settings, this root is very close to unity so that shocks die out rather slowly.

The GARCH(q, p) Model

Higher order GARCH models, denoted GARCH(q, p), can be estimated by choosing either q or p greater than 1 where q is the order of the autoregressive GARCH terms and p is the order of the moving average ARCH terms.

The representation of the GARCH(q, p) variance is:

$$\sigma_t^2 = \omega + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2 + \sum_{i=1}^{p} \alpha_i \epsilon_{t-i}^2 \tag{20.5}$$

The GARCH-M Model

The X_t in equation (20.2) represent exogenous or predetermined variables that are included in the mean equation. If we introduce the conditional variance or standard deviation into the mean equation, we get the GARCH-in-Mean (GARCH-M) model (Engle, Lilien and Robins, 1987):

$$Y_t = X_t'\theta + \lambda \sigma_t^2 + \epsilon_t . \tag{20.6}$$

The ARCH-M model is often used in financial applications where the expected return on an asset is related to the expected asset risk. The estimated coefficient on the expected risk is a measure of the risk-return tradeoff.

Two variants of this ARCH-M specification use the conditional standard deviation or the log of the conditional variance in place of the variance in Equation (20.6).

$$Y_t = X_t'\theta + \lambda \sigma_t + \epsilon_t . \tag{20.7}$$

$$Y_t = X_t'\theta + \lambda \log(\sigma_t^2) + \epsilon_t \tag{20.8}$$

Regressors in the Variance Equation

Equation (20.5) may be extended to allow for the inclusion of exogenous or predetermined regressors, z, in the variance equation:

$$\sigma_t^2 = \omega + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2 + \sum_{i=1}^{p} \alpha_i \epsilon_{t-i}^2 + Z_t'\pi . \tag{20.9}$$

Note that the forecasted variances from this model are not guaranteed to be positive. You may wish to introduce regressors in a form where they are always positive to minimize the possibility that a single, large negative value generates a negative forecasted value.

Distributional Assumptions

To complete the basic ARCH specification, we require an assumption about the conditional distribution of the error term ϵ. There are three assumptions commonly employed when working with ARCH models: normal (Gaussian) distribution, Student's t-distribution, and the Generalized Error Distribution (GED). Given a distributional assumption, ARCH models are typically estimated by the method of maximum likelihood.

For example, for the GARCH(1, 1) model with conditionally normal errors, the contribution to the log-likelihood for observation t is:

$$l_t = -\frac{1}{2}\log(2\pi) - \frac{1}{2}\log\sigma_t^2 - \frac{1}{2}(y_t - X_t'\theta)^2/\sigma_t^2, \tag{20.10}$$

where σ_t^2 is specified in one of the forms above.

For the Student's t-distribution, the log-likelihood contributions are of the form:

$$l_t = -\frac{1}{2}\log\left(\frac{\pi(\nu-2)\Gamma(v/2)^2}{\Gamma((v+1)/2)^2}\right) - \frac{1}{2}\log\sigma_t^2 - \frac{(\nu+1)}{2}\log\left(1 + \frac{(y_t - X_t'\theta)^2}{\sigma_t^2(\nu-2)}\right) \tag{20.11}$$

where the degree of freedom $\nu > 2$ controls the tail behavior. The t-distribution approaches the normal as $\nu \to \infty$.

For the GED, we have:

$$l_t = -\frac{1}{2}\log\left(\frac{\Gamma(1/r)^3}{\Gamma(3/r)(r/2)^2}\right) - \frac{1}{2}\log\sigma_t^2 - \left(\frac{\Gamma(3/r)(y_t - X_t'\theta)^2}{\sigma_t^2\Gamma(1/r)}\right)^{r/2} \tag{20.12}$$

where the tail parameter $r > 0$. The GED is a normal distribution if $r = 2$, and fat-tailed if $r < 2$.

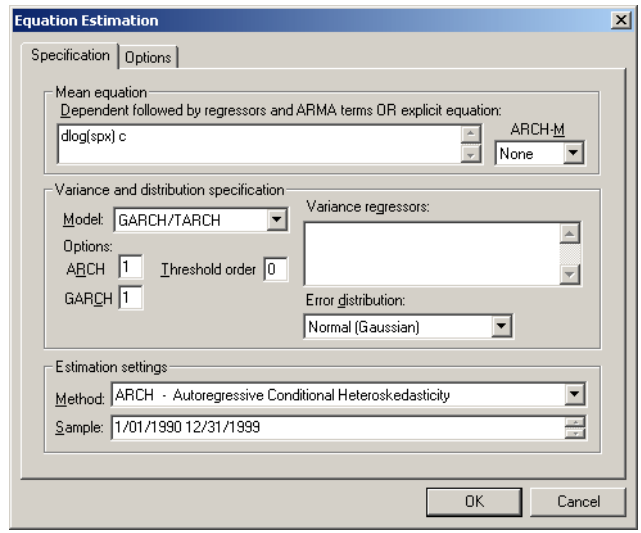
By default, ARCH models in EViews are estimated by the method of maximum likelihood under the assumption that the errors are conditionally normally distributed.

Estimating ARCH Models in EViews

To estimate an ARCH or GARCH model, open the equation specification dialog by selecting **Quick/Estimate Equation...** or by selecting **Object/New Object.../Equation...**. Select **ARCH** from the method combo box at the bottom of the dialog.

The dialog will change to show you the ARCH specification dialog. You will need to specify both the mean and the variance specifications, the error distribution and the estimation sample.

The Mean Equation



In the dependent variable edit box, you should enter the specification of the mean equation. You can enter the specification in list form by listing the dependent variable followed by the regressors. You should add the C to your specification if you wish to include a constant. If you have a more complex mean specification, you can enter your mean equation using an explicit expression.

If your specification includes an ARCH-M term, you should select the appropriate item of the combo box in the upper right-hand side of the dialog.

The Variance Equation

Your next step is to specify your variance equation.

Class of models

To estimate one of the standard GARCH models as described above, select the **GARCH/TARCH** entry from the **Model** combo box. The other entries (**EGARCH**, **PARCH**, and **Component ARCH(1, 1)**) correspond to more complicated variants of the GARCH specification. We discuss each of these models in "Additional ARCH Models" on page 612.

Under the **Options** label, you should choose the number of ARCH and GARCH terms. The default, which includes one ARCH and one GARCH term is by far the most popular specification.

If you wish to estimate an asymmetric model, you should enter the number of asymmetry terms in the **Threshold orde**r edit field. The default settings estimate a symmetric model.

Variance regressors

In the **Variance Regressors** edit box, you may optionally list variables you wish to include in the variance specification. Note that EViews will always include a constant as a variance regressor so that you do not need to add C to the list.

The distinction between the permanent and transitory regressors is discussed in "The Component GARCH (CGARCH) Model" on page 615.

The Error Distribution

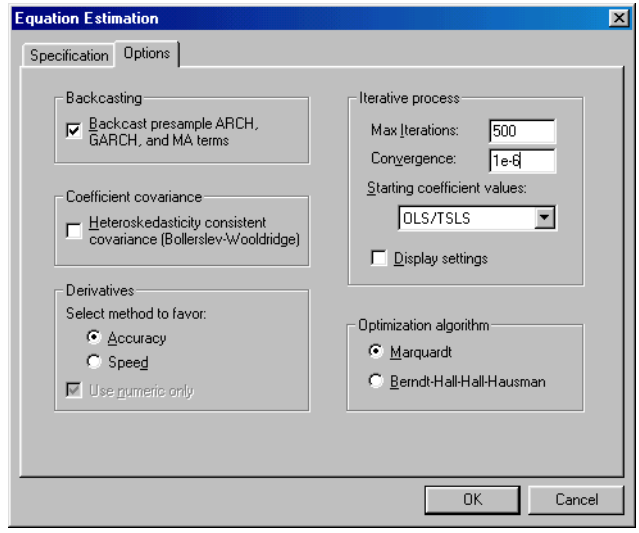
To specify the form of the conditional distribution for your errors, you should select an entry from the combo box labeled **Error Distribution**. You may choose between the default **Normal (Gaussian)**, the **Student's t**, the **Generalized Error (GED)**, the **Student's t with fixed d.f.**, or the **GED with fixed parameter**. In the latter two cases, you will be prompted to enter a value for the fixed parameter. See "Distributional Assumptions" on page 604 for details on the supported distributions.

Estimation Options

EViews provides you with access to a number of optional estimation settings. Simply click on the **Options** tab and fill out the dialog as desired.

Backcasting

By default, both the innovations used in initializing MA estimation and the initial variance required for the GARCH terms are computed using backcasting methods. Details on the MA backcasting procedure are provided in "Backcasting MA terms" on page 510.



When computing backcast initial variances for GARCH, EViews first uses the coefficient values to compute the residuals of the mean equation, and then computes an exponential smoothing estimator of the initial values,

$$\sigma_0^2 = \epsilon_0^2 = \lambda^T \hat{\sigma}^2 + (1-\lambda) \sum_{j=0}^{T} \lambda^{T-j-1} (\hat{\epsilon}_{T-j}^2), \tag{20.13}$$

where $\hat{\epsilon}$ are the residuals from the mean equation, $\hat{\sigma}^2$ is the unconditional variance estimate:

$$\hat{\sigma}^2 = \sum_{t=1}^{T} \hat{\epsilon}_t^2 / T \tag{20.14}$$

and the smoothing parameter $\lambda = 0.7$. Alternatively, you can choose to initialize the GARCH process using the unconditional variance:

$$\sigma_0^2 = \hat{\sigma}^2. \tag{20.15}$$

If you turn off backcasting, EViews will set the presample values of the residual to zero to initialize an MA, if present, and will set the presample values of the variance to the unconditional variance using Equation (20.15).

Our experience has been that GARCH models initialized using backcast exponential smoothing often outperform models initialized using the unconditional variance.

Heteroskedasticity Consistent Covariances

Click on the check box labeled **Heteroskedasticity Consistent Covariance** to compute the quasi-maximum likelihood (QML) covariances and standard errors using the methods described by Bollerslev and Wooldridge (1992). This option is only available if you choose the conditional normal as the error distribution.

You should use this option if you suspect that the residuals are not conditionally normally distributed. When the assumption of conditional normality does not hold, the ARCH parameter estimates will still be consistent, provided the mean and variance functions are correctly specified. The estimates of the covariance matrix will not be consistent unless this option is specified, resulting in incorrect standard errors.

Note that the parameter estimates will be unchanged if you select this option; only the estimated covariance matrix will be altered.

Derivative Methods

EViews currently uses numeric derivatives in estimating ARCH models. You can control the method used in computing these derivatives to favor speed (fewer function evaluations) or to favor accuracy (more function evaluations).

Iterative Estimation Control

The likelihood functions of ARCH models are not always well-behaved so that convergence may not be achieved with the default estimation settings. You can use the options dialog to select the iterative algorithm (Marquardt, BHHH/Gauss-Newton), change starting values, increase the maximum number of iterations, or adjust the convergence criterion.

Starting Values

As with other iterative procedures, starting coefficient values are required. EViews will supply its own starting values for ARCH procedures using OLS regression for the mean equation. Using the Options dialog, you can also set starting values to various fractions of

the OLS starting values, or you can specify the values yourself by choosing the User Specified option, and placing the desired coefficients in the default coefficient vector.

GARCH(1,1) examples

To estimate a standard GARCH(1,1) model with no regressors in the mean and variance equations:

$$\begin{aligned} R_t &= c + \epsilon_t \\ \sigma_t^2 &= \omega + \alpha \epsilon_{t-1}^2 + \beta \sigma_{t-1}^2 \end{aligned} \tag{20.16}$$

you should enter the various parts of your specification:

- Fill in the Mean Equation Specification edit box as

```
r c
```

- Enter `1` for the number of ARCH terms, and `1` for the number of GARCH terms, and select **GARCH/TARCH**.
- Select **None** for the **ARCH-M term**.
- Leave blank the **Variance Regressors** edit box.

To estimate the ARCH(4)-M model:

$$\begin{aligned} R_t &= \gamma_0 + \gamma_1 DUM_t + \gamma_2 \sigma_t + \epsilon_t \\ \sigma_t^2 &= \omega + \alpha_1 \epsilon_{t-1}^2 + \alpha_2 \epsilon_{t-2}^2 + \alpha_3 \epsilon_{t-3}^2 + \alpha_4 \epsilon_{t-4}^2 + \gamma_3 DUM_t \end{aligned} \tag{20.17}$$

you should fill out the dialog in the following fashion:

- Enter the mean equation specification "R C DUM".
- Enter "4" for the ARCH term and "0" for the GARCH term, and select **GARCH (symmetric)**.
- Select **Std. Dev.** for the **ARCH-M** term.
- Enter DUM in the Variance Regressors edit box.

Once you have filled in the Equation Specification dialog, click **OK** to estimate the model. ARCH models are estimated by the method of maximum likelihood, under the assumption that the errors are conditionally normally distributed. Because the variance appears in a non-linear way in the likelihood function, the likelihood function must be estimated using iterative algorithms. In the status line, you can watch the value of the likelihood as it changes with each iteration. When estimates converge, the parameter estimates and conventional regression statistics are presented in the ARCH object window.

As an example, we fit a GARCH(1,1) model to the first difference of log daily S&P 500 (DLOG(SPX)) using backcast values for the initial variances and Bollerslev-Wooldridge standard errors. The output is presented below:

Dependent Variable: DLOG(SPX)
Method: ARCH - Normal distribution (Marquardt)
Date: 09/16/03 Time: 13:52
Sample: 1/02/1990 12/31/1999
Included observations: 2528
Convergence achieved after 12 iterations
Bollerslev-Wooldrige robust standard errors & covariance
Variance backcast: ON
GARCH = C(2) + C(3)*RESID(-1)^2 + C(4)*GARCH(-1)

	Coefficient	Std. Error	z-Statistic	Prob.
C	0.000598	0.000143	4.176757	0.0000
	Variance Equation			
C	5.82E-07	1.93E-07	3.018858	0.0025
RESID(-1)^2	0.053337	0.011688	4.563462	0.0000
GARCH(-1)	0.939945	0.011201	83.91679	0.0000

R-squared	-0.000015	Mean dependent var	0.000564
Adjusted R-squared	-0.001204	S.D. dependent var	0.008888
S.E. of regression	0.008894	Akaike info criterion	-6.807476
Sum squared resid	0.199649	Schwarz criterion	-6.798243
Log likelihood	8608.650	Durbin-Watson stat	1.964028

By default, the estimation output header describes the estimation sample, and the methods used for computing the coefficient standard errors, the initial variance terms, and the variance equation.

The main output from ARCH estimation is divided into two sections—the upper part provides the standard output for the mean equation, while the lower part, labeled "Variance Equation", contains the coefficients, standard errors, z-statistics and p-values for the coefficients of the variance equation.

The ARCH parameters correspond to α and the GARCH parameters to β in Equation (20.2) on page 601. The bottom panel of the output presents the standard set of regression statistics using the residuals from the mean equation. Note that measures such as R^2 may not be meaningful if there are no regressors in the mean equation. Here, for example, the R^2 is negative.

In this example, the sum of the ARCH and GARCH coefficients ($\alpha + \beta$) is very close to one, indicating that volatility shocks are quite persistent. This result is often observed in high frequency financial data.

Working with ARCH Models

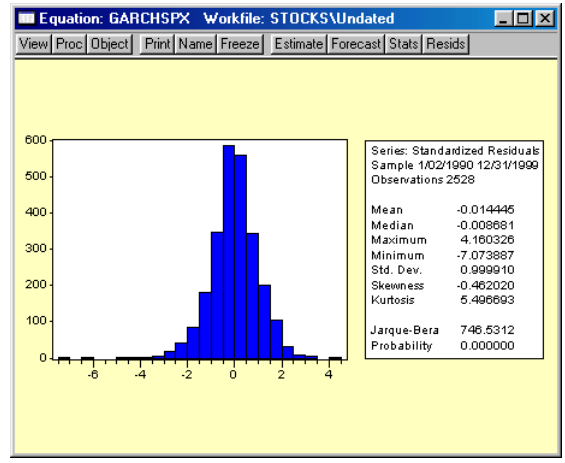
Once your model has been estimated, EViews provides a variety of views and procedures for inference and diagnostic checking.

Views of ARCH Models

- **Actual, Fitted, Residual** view displays the residuals in various forms, such as table, graphs, and standardized residuals. You can save the residuals as a named series in your workfile using a procedure (see below).
- **GARCH Graph** plots the one-step ahead standard deviation σ_t or variance σ_t^2 for each observation in the sample. The observation at period t is the forecast for t made using information available in $t-1$. You can save the conditional standard deviations or variances as named series in your workfile using a procedure (see below). If the specification is for a component model, EViews will also display the permanent and transitory components.
- **Covariance Matrix** displays the estimated coefficient covariance matrix. Most ARCH models (except ARCH-M models) are block diagonal so that the covariance between the mean coefficients and the variance coefficients is very close to zero. If you include a constant in the mean equation, there will be two C's in the covariance matrix; the first C is the constant of the mean equation, and the second C is the constant of the variance equation.
- **Coefficient Tests** carries out standard hypothesis tests on the estimated coefficients. See "Coefficient Tests" on page 570 for details. Note that the likelihood ratio tests are not appropriate under a quasi-maximum likelihood interpretation of your results.
- **Residual Tests/Correlogram–Q-statistics** displays the correlogram (autocorrelations and partial autocorrelations) of the standardized residuals. This view can be used to test for remaining serial correlation in the mean equation and to check the specification of the mean equation. If the mean equation is correctly specified, all Q-statistics should not be significant. See "Correlogram" on page 326 for an explanation of correlograms and Q-statistics.
- **Residual Tests/Correlogram Squared Residuals** displays the correlogram (autocorrelations and partial autocorrelations) of the squared standardized residuals. This view can be used to test for remaining ARCH in the variance equation and to check the specification of the variance equation. If the variance equation is correctly specified, all Q-statistics should not be significant. See "Correlogram" on page 326 for an

explanation of correlograms and Q-statistics. See also **Residual Tests/ARCH LM Test**.

- **Residual Tests/Histogram–Normality Test** displays descriptive statistics and a histogram of the standardized residuals. You can use the Jarque-Bera statistic to test the null of whether the standardized residuals are normally distributed. If the standardized residuals are normally distributed, the Jarque-Bera statistic should not be significant. See "Descriptive Statistics" beginning on page 310 for an explanation of the Jarque-Bera test. For example, the histogram of the standardized residuals from the GARCH(1,1) model fit to the daily stock return looks as follows:



  The standardized residuals are leptokurtic and the Jarque-Bera statistic strongly rejects the hypothesis of normal distribution.

- **Residual Tests/ARCH LM Test** carries out Lagrange multiplier tests to test whether the standardized residuals exhibit additional ARCH. If the variance equation is correctly specified, there should be no ARCH left in the standardized residuals. See "ARCH LM Test" on page 582 for a discussion of testing. See also **Residual Tests/Correlogram Squared Residuals.**

ARCH Model Procedures

- **Make Residual Series** saves the residuals as named series in your workfile. You have the option to save the ordinary residuals, ϵ_t, or the standardized residuals, ϵ_t/σ_t. The residuals will be named RESID1, RESID2, and so on; you can rename the series with the **name** button in the series window.

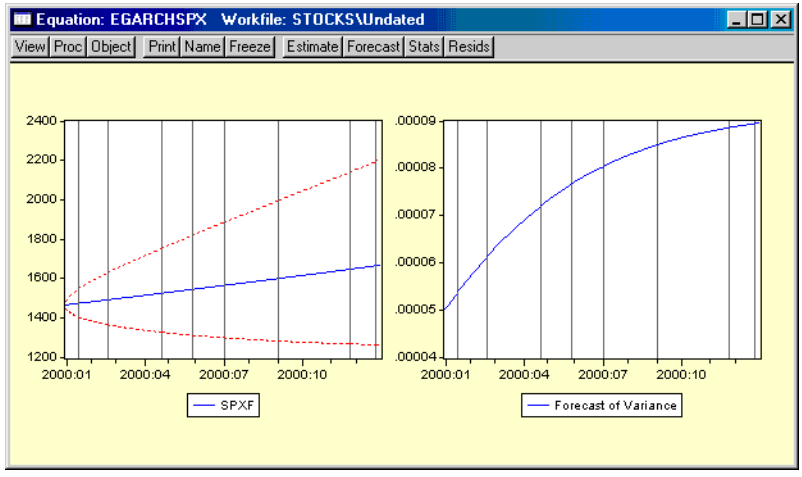
- **Make GARCH Variance Series...** saves the conditional variances σ_t^2 as named series in your workfile. You should provide a name for the target conditional variance series and, if relevant, you may provide a name for the permanent component series. You may take the square root of the conditional variance series to get the conditional standard deviations as displayed by the **View/GARCH Graph/Conditional Standard Deviation.**

- **Forecast** uses the estimated ARCH model to compute static and dynamic forecasts of the mean, its forecast standard error, and the conditional variance. To save any of

these forecasts in your workfile, type a name in the corresponding dialog box. If you choose the **Do graph** option, EViews displays the graphs of the forecasts and two standard deviation bands for the mean forecast.

Note that the squared residuals ϵ_t^2 may not be available for presample values or when computing dynamic forecasts. In such cases, EViews will replaced the term by its expect value. In the simple GARCH(p, q) case, for example, the expected value of the squared residual is the fitted variance, *e.g.*, $E(\epsilon_t^2) = \sigma_t^2$. In other models, the expected value of the residual term will differ depending on the distribution and, in some cases, the estimated parameters of the model.

For example, to construct dynamic forecasts of SPX using the previously estimated model, click on **Forecast** and fill in the **Forecast** dialog, setting the sample after the estimation period. If you choose **Do graph**, the equation view changes to display the forecast results. Here, we compute the forecasts from Jan. 1, 2000 to Jan. 1, 2001, and display them side-by-side.



The first graph is the forecast of SPX (SPXF) from the mean equation with two standard deviation bands. The second graph is the forecast of the conditional variance σ_t^2.

Additional ARCH Models

In addition to the standard GARCH specification, EViews has the flexibility to estimate several other variance models. These include TARCH, EGARCH, PARCH, and component GARCH. For each of these models, the user has the ability to choose the order, if any, of asymmetry.

The Threshold GARCH (TARCH) Model

TARCH or Threshold ARCH and Threshold GARCH were introduced independently by Zakoïan (1994) and Glosten, Jaganathan, and Runkle (1993). The generalized specification for the conditional variance is given by:

$$\sigma_t^2 = \omega + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2 + \sum_{i=1}^{p} \alpha_i \epsilon_{t-i}^2 + \sum_{k=1}^{r} \gamma_k \epsilon_{t-k}^2 I_{t-k}^- \tag{20.18}$$

where $I_t^- = 1$ if $\epsilon_t < 0$ and 0 otherwise.

In this model, good news, $\epsilon_{t-i} > 0$, and bad news. $\epsilon_{t-i} < 0$, have differential effects on the conditional variance; good news has an impact of α_i, while bad news has an impact of $\alpha_i + \gamma_i$. If $\gamma_i > 0$, bad news increases volatility, and we say that there is a *leverage effect* for the i-th order. If $\gamma_i \neq 0$, the news impact is asymmetric.

Note that GARCH is a special case of the TARCH model where the threshold term is set to zero. To estimate a TARCH model, specify your GARCH model with ARCH and GARCH order and then change the **Threshold order** to the desired value.

The Exponential GARCH (EGARCH) Model

The EGARCH or Exponential GARCH model was proposed by Nelson (1991). The specification for the conditional variance is:

$$\log(\sigma_t^2) = \omega + \sum_{j=1}^{q} \beta_j \log(\sigma_{t-j}^2) + \sum_{i=1}^{p} \alpha_i \left| \frac{\epsilon_{t-i}}{\sigma_{t-i}} \right| + \sum_{k=1}^{r} \gamma_k \frac{\epsilon_{t-k}}{\sigma_{t-k}}. \tag{20.19}$$

Note that the left-hand side is the *log* of the conditional variance. This implies that the leverage effect is exponential, rather than quadratic, and that forecasts of the conditional variance are guaranteed to be nonnegative. The presence of leverage effects can be tested by the hypothesis that $\gamma_i < 0$. The impact is asymmetric if $\gamma_i \neq 0$.

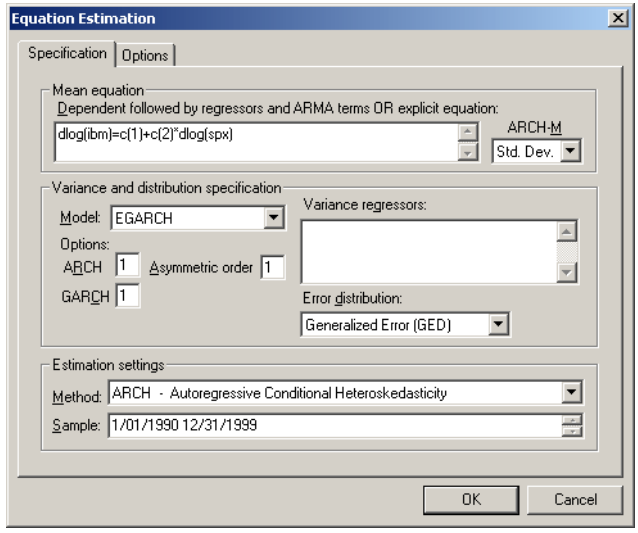
There are a couple of differences between the EViews specification of the EGARCH model and the original Nelson model. First, Nelson assumes that the ϵ_t follows a Generalized Error Distribution (GED), while EViews gives you a choice of normal, Student's t-distribution, or GED. Second, Nelson's specification for the log conditional variance is a restricted version of:

$$\log(\sigma_t^2) = \omega + \sum_{j=1}^{q} \beta_j \log(\sigma_{t-j}^2) + \sum_{i=1}^{p} \alpha_i \left| \frac{\epsilon_{t-i}}{\sigma_{t-i}} - E\left(\frac{\epsilon_{t-i}}{\sigma_{t-i}} \right) \right| + \sum_{k=1}^{r} \gamma_k \frac{\epsilon_{t-k}}{\sigma_{t-k}}$$

which differs slightly from the specification above. Estimating this model will yield identical estimates to those reported by EViews except for the intercept term w, which will dif-

fer in a manner that depends upon the distributional assumption and the order p. For example, in a $p = 1$ model with a normal distribution, the difference will be $\alpha_1\sqrt{2/\pi}$.

To estimate an EGARCH model, simply select the **EGARCH** in the model specification combo box and enter the orders for the **ARCH**, **GARCH** and the **Asymmetry order**.



The Power ARCH (PARCH) Model

Taylor (1986) and Schwert (1989) introduced the standard deviation GARCH model, where the standard deviation is modeled rather than the variance. This model, along with several other models, is generalized in Ding *et al.* (1993) with the Power ARCH specification. In the Power ARCH model, the power parameter δ of the standard deviation can be estimated rather than imposed, and the optional γ parameters are added to capture asymmetry of up to order r:

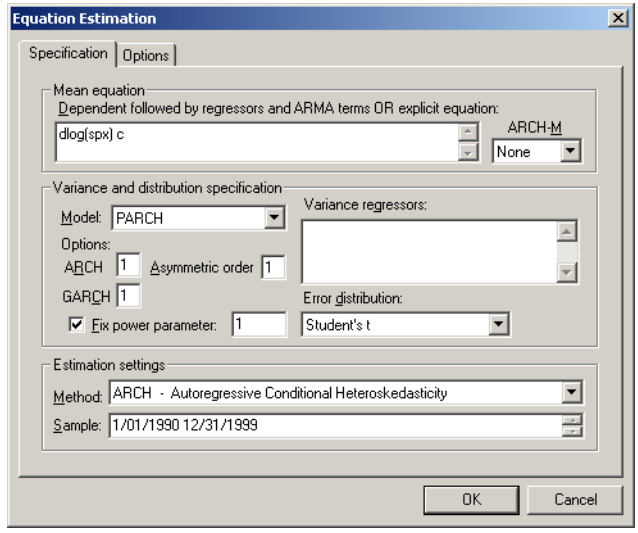
$$\sigma_t^{\delta} = \omega + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^{\delta} + \sum_{i=1}^{p} \alpha_i (|\epsilon_{t-i}| - \gamma_i \epsilon_{t-i})^{\delta} \tag{20.20}$$

where $\delta > 0$, $|\gamma_i| \leq 1$ for $i = 1, \ldots, r$, $\gamma_i = 0$ for all $i > r$, and $r \leq p$.

The symmetric model sets $\gamma_i = 0$ for all i. Note that if $\delta = 2$ and $\gamma_i = 0$ for all i, the PARCH model is simply a standard GARCH specification. As in the previous models, the asymmetric effects are present if $\gamma \neq 0$.

To estimate this model, simply select the PARCH in the model specification combo box and input the orders for the **ARCH**, **GARCH** and **Asymmetric** terms. EViews provides you with the option of either estimating or fixing a value for δ. To estimate the Taylor-Schwert's

model, for example, you will to set the order of the asymmetric terms to zero and will set δ to 1.



The Component GARCH (CGARCH) Model

The conditional variance in the GARCH(1, 1) model:

$$\sigma_t^2 = \bar{\omega} + \alpha(\epsilon_{t-1}^2 - \bar{\omega}) + \beta(\sigma_{t-1}^2 - \bar{\omega}). \tag{20.21}$$

shows mean reversion to $\bar{\omega}$, which is a constant for all time. By contrast, the component model allows mean reversion to a varying level m_t, modeled as:

$$\begin{aligned} \sigma_t^2 - m_t &= \bar{\omega} + \alpha(\epsilon_{t-1}^2 - \bar{\omega}) + \beta(\sigma_{t-1}^2 - \bar{\omega}) \\ m_t &= \omega + \rho(m_{t-1} - \omega) + \phi(\epsilon_{t-1}^2 - \sigma_{t-1}^2). \end{aligned} \tag{20.22}$$

Here σ_t^2 is still the volatility, while q_t takes the place of ω and is the time varying long-run volatility. The first equation describes the transitory component, $\sigma_t^2 - q_t$, which converges to zero with powers of $(\alpha + \beta)$. The second equation describes the long run component m_t, which converges to ω with powers of ρ. ρ is typically between 0.99 and 1 so that m_t approaches ω very slowly. We can combine the transitory and permanent equations and write:

$$\begin{aligned} \sigma_t^2 = (1-\alpha-\beta)(1-\rho)\omega + (\alpha+\phi)\epsilon_{t-1}^2 - (\alpha\rho + (\alpha+\beta)\phi)\epsilon_{t-2}^2 \\ + (\beta-\phi)\sigma_{t-1}^2 - (\beta\rho - (\alpha+\beta)\phi)\sigma_{t-2}^2 \end{aligned} \tag{20.23}$$

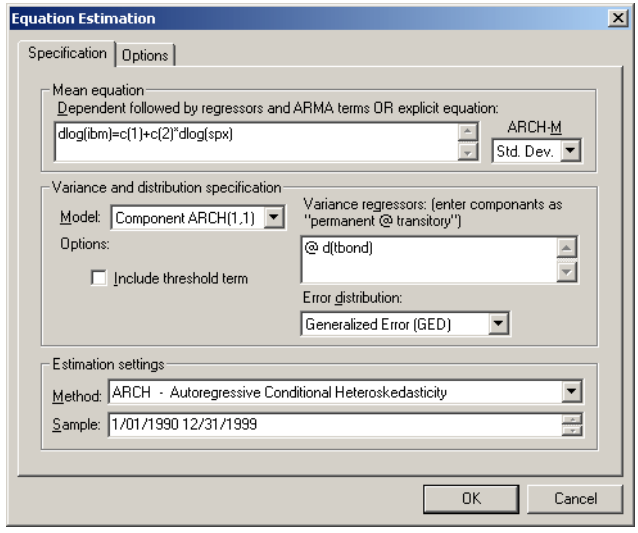
which shows that the component model is a (nonlinear) restricted GARCH(2, 2) model.

To select the Component ARCH model, simply choose **Component ARCH(1,1)** in the **Model** combo box. You can include exogenous variables in the conditional variance equation of component models, either in the permanent or transitory equation (or both). The variables in the transitory equation will have an impact on the short run movements in volatility, while the variables in the permanent equation will affect the long run levels of volatility.

An asymmetric Component ARCH model may be estimated by checking the **Include threshold term** checkbox. This option combines the component model with the asymmetric TARCH model, introducing asymmetric effects in the transitory equation and estimates models of the form:

$$\begin{aligned} y_t &= x_t'\pi + \epsilon_t \\ m_t &= \omega + \rho(m_{t-1} - \omega) + \phi(\epsilon_{t-1}^2 - \sigma_{t-1}^2) + \theta_1 z_{1t} \\ \sigma_t^2 - m_t &= \alpha(\epsilon_{t-1}^2 - m_{t-1}) + \gamma(\epsilon_{t-1}^2 - m_{t-1})d_{t-1} + \beta(\sigma_{t-1}^2 - m_{t-1}) + \theta_2 z_{2t} \end{aligned} \tag{20.24}$$

where z are the exogenous variables and d is the dummy variable indicating negative shocks. $\gamma > 0$ indicates the presence of transitory leverage effects in the conditional variance.

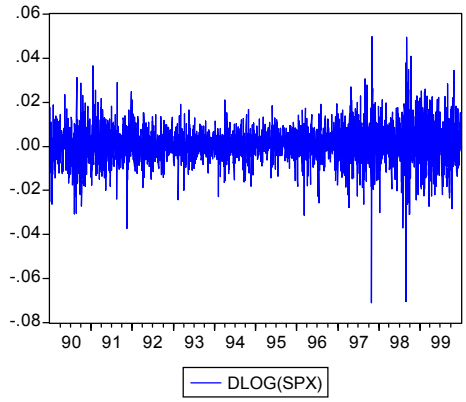


User Specified Models

In some cases, you might wish to estimate an ARCH model not mentioned above, for example a special variant of PARCH. Many other ARCH models can be estimated using the logl object. For example, Chapter 22, "The Log Likelihood (LogL) Object", beginning on page 671 contains examples of using logl objects for simple bivariate GARCH models.

Examples

As an illustration of ARCH modeling in EViews, we estimate a model for the daily S&P 500 stock index from 1990 to 1999. The dependent variable is the daily continuously compounding return, $\log(s_t/s_{t-1})$, where s_t is the daily close of the index. A graph of the return series clearly shows volatility clustering.



We will specify our mean equation with a simple constant:

$$\log(s_t/s_{t-1}) = c_1 + \epsilon_t \qquad (20.25)$$

For the variance specification, we employ an EGARCH(1, 1) model:

$$\log(\sigma_t^2) = \omega + \beta\log(\sigma_{t-1}^2) + \alpha\left|\frac{\epsilon_{t-1}}{\sigma_{t-1}}\right| + \gamma\frac{\epsilon_{t-1}}{\sigma_{t-1}} \qquad (20.26)$$

When we previously estimated a GARCH(1,1) model with the data, the standardized residual showed evidence of excess kurtosis. To model the thick tail in the residuals, we will assume that the errors follow a Student's t-distribution.

To estimate this model, open the GARCH estimation dialog, enter the mean specification:

```
dlog(spx) c
```

select the **EGARCH** method, enter 1 for the **ARCH** and **GARCH** orders and the **Asymmetric order**, and select **Student's t** for the **Error distribution**. Click on **OK** to continue.

EViews displays the results of the estimation procedure. The top portion contains a description of the estimation specification, including the estimation sample, error distribution assumption, and backcast assumption.

Below the header information are the results for the mean and the variance equations, followed by the results for any distributional parameters. Here, we see that the relatively small degrees of freedom parameter for the t-distributiont suggests that the distribution of the standardized errors departs significantly from normality.

Method: ARCH - Student's t distribution (Marquardt)
Date: 09/16/03 Time: 13:52
Sample: 1/02/1990 12/31/1999
Included observations: 2528
Convergence achieved after 18 iterations
Variance backcast: ON
LOG(GARCH) = C(2) + C(3)*ABS(RESID(-1)/@SQRT(GARCH(-1))) + C(4)*RESID(-1)/@SQRT(GARCH(-1)) + C(5)*LOG(GARCH(-1))

	Coefficient	Std. Error	z-Statistic	Prob.
C	0.000513	0.000135	3.810316	0.0001
Variance Equation				
C(2)	-0.196806	0.039216	-5.018500	0.0000
C(3)	0.113679	0.017573	6.468881	0.0000
C(4)	-0.064136	0.011584	-5.536811	0.0000
C(5)	0.988574	0.003366	293.7065	0.0000
T-DIST. DOF	6.703937	0.844864	7.934935	0.0000

R-squared	-0.000032	Mean dependent var	0.000564
Adjusted R-squared	-0.002015	S.D. dependent var	0.008888
S.E. of regression	0.008897	Akaike info criterion	-6.871798
Sum squared resid	0.199653	Schwarz criterion	-6.857949
Log likelihood	8691.953	Durbin-Watson stat	1.963994

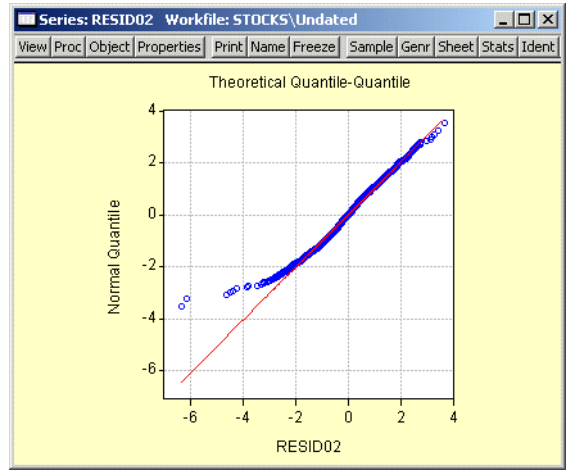
To test whether there any remaining ARCH effects in the residuals, select **View/Residual Tests/ARCH LM Test...** and specify the order to test. The top portion of the output from testing up-to an ARCH(7) is given by:

ARCH Test:

F-statistic	0.398638	Probability	0.903581
Obs*R-squared	2.796245	Probability	0.903190

so there is little evidence of remaining ARCH effects.

One way of further examining the distribution of the residuals is to plot the quantiles. First, save the standardized residuals by clicking on **Proc/Make Residual Series...**, select the **Standardized** option, and specify a name for the resulting series. EViews will create a series containing the desired residuals; in this example, we create a series named RESID02. Then open the residual series window and select **View/Distribution/Quantile-Quantile Graphs...** and tell EViews the distribution whose quantiles you wish to plot against, for example, the **Normal distribution**.

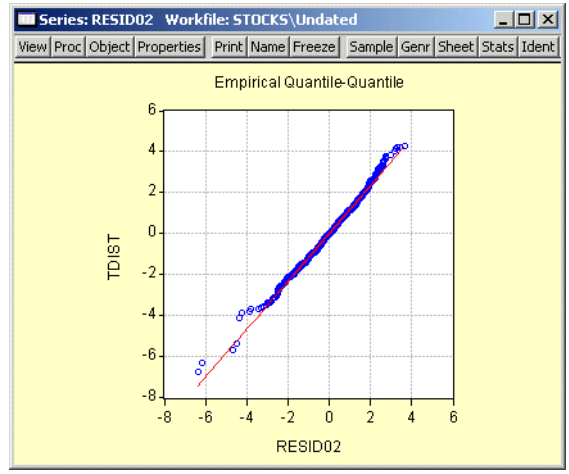


If the residuals are normally distributed, the points in the QQ-plots should lie alongside a straight line; see "Quantile-Quantile" on page 393 for details on QQ-plots. The plot indicates that it is primarily large negative shocks that are driving the departure from normality. Note that we have modified the QQ-plot slightly by setting identical axes to facilitate comparison with the diagonal line.

We can also plot the residuals against the quantiles of the *t*-distribution. Although there is no option for the t-distribution in the Quantile-Quantile plot view, you may simulate a draw from a *t*-distribution and examine whether the quantiles of the simulated observations match the quantiles of the residuals. The command:

```
series tdist = @qtdist(rnd, 6.7)
```

simulates a random draw from the *t*-distribution with 6.7 degrees of freedom. Then, in the **QQ Plot** dialog click on the **Series or Group** radio button and type the name of a series (in this case "TDIST").

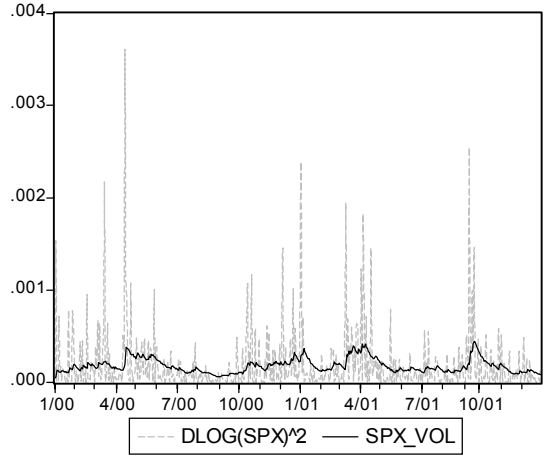


The large negative residuals more closely follow a straight line. On the other hand, one can see a slight deviation from *t*-distribution for large positive shocks. This is not unexpected, as the previous QQ-plot suggested that, with the exception of the large negative shocks, the residuals were close to normally distributed.

To see how the model might fit real data, we examine static forecasts for out-of-sample data. Click on **Forecast**, type in SPX_VOL in the GARCH field to save the fore-

casted conditional variance, change the sample to the post-estimation sample period "1/1/2000 1/1/2002" and click on **Static** to select a static forecast.

Since the actual volatility is unobserved, we will use the squared return series (DLOG(SPX)^2) as a proxy for the realized volatility. A plot of the proxy against the forecasted volatility provides an indication of the model's ability to track variations in market volatility.



Chapter 21. Discrete and Limited Dependent Variable Models

The regression methods described in Chapter 15, "Basic Regression" and Chapter 16, "Additional Regression Methods" require that the dependent variable be observed on a continuous and unrestricted scale. It is quite common, however, for this condition to be violated, resulting in a non-continuous, or a limited dependent variable. We will distinguish between three types of these variables:

- qualitative (observed on a discrete or ordinal scale)
- censored or truncated
- integer valued

In this chapter, we discuss estimation methods for several qualitative and limited dependent variable models. EViews provides estimation routines for binary or ordered (probit, logit, gompit), censored or truncated (tobit, etc.), and integer valued (count data) models.

Standard introductory discussion for the models presented in this chapter may be found in Greene (1997), Johnston and DiNardo (1997), and Maddala (1983). Wooldridge (1996) provides an excellent reference for quasi-likelihood methods and count models.

Binary Dependent Variable Models

In this class of models, the dependent variable, y may take on only two values— y might be a dummy variable representing the occurrence of an event, or a choice between two alternatives. For example, you may be interested in modeling the employment status of each individual in your sample (whether employed or not). The individuals differ in age, educational attainment, race, marital status, and other observable characteristics, which we denote as x. The goal is to quantify the relationship between the individual characteristics and the probability of being employed.

Theory

Suppose that a binary dependent variable, y, takes on values of zero and one. A simple linear regression of y on x is not appropriate, since among other things, the implied model of the conditional mean places inappropriate restrictions on the residuals of the model. Furthermore, the fitted value of y from a simple linear regression is not restricted to lie between zero and one.

Instead, we adopt a specification that is designed to handle the specific requirements of binary dependent variables. Suppose that we model the probability of observing a value of one as:

$$\Pr(y_i = 1 | x_i, \beta) = 1 - F(-x_i'\beta), \tag{21.1}$$

where F is a continuous, strictly increasing function that takes a real value and returns a value ranging from zero to one. The choice of the function F determines the type of binary model. It follows that:

$$\Pr(y_i = 0 | x_i, \beta) = F(-x_i'\beta). \tag{21.2}$$

Given such a specification, we can estimate the parameters of this model using the method of maximum likelihood. The likelihood function is given by:

$$l(\beta) = \sum_{i=0}^{n} y_i \log(1 - F(-x_i'\beta)) + (1 - y_i)\log(F(-x_i'\beta)). \tag{21.3}$$

The first order conditions for this likelihood are nonlinear so that obtaining parameter estimates requires an iterative solution. By default, EViews uses a second derivative method for iteration and computation of the covariance matrix of the parameter estimates. As discussed below, EViews allows you to override these defaults using the Options dialog (see "Second Derivative Methods" on page 956 for additional details on the estimation methods).

There are two alternative interpretations of this specification that are of interest. First, the binary model is often motivated as a latent variables specification. Suppose that there is an unobserved latent variable y_i^* that is linearly related to x:

$$y_i^* = x_i'\beta + u_i \tag{21.4}$$

where u_i is a random disturbance. Then the observed dependent variable is determined by whether y_i^* exceeds a threshold value:

$$y_i = \begin{cases} 1 & \text{if } y_i^* > 0 \\ 0 & \text{if } y_i^* \le 0. \end{cases} \tag{21.5}$$

In this case, the threshold is set to zero, but the choice of a threshold value is irrelevant, so long as a constant term is included in x_i. Then :

$$\Pr(y_i = 1 | x_i, \beta) = \Pr(y_i^* > 0) = \Pr(x_i'\beta + u_i > 0) = 1 - F_u(-x_i'\beta) \tag{21.6}$$

where F_u is the cumulative distribution function of u. Common models include probit (standard normal), logit (logistic), and gompit (extreme value) specifications for the F function.

In principle, the coding of the two numerical values of y is not critical since each of the binary responses only represents an event. Nevertheless, EViews requires that you code y as a zero-one variable. This restriction yields a number of advantages. For one, coding the

variable in this fashion implies that expected value of y is simply the probability that $y = 1$:

$$\begin{aligned} \mathrm{E}(y_i | x_i, \beta) &= 1 \cdot \Pr(y_i = 1 | x_i, \beta) + 0 \cdot \Pr(y_i = 0 | x_i, \beta) \\ &= \Pr(y_i = 1 | x_i, \beta). \end{aligned} \tag{21.7}$$

This convention provides us with a second interpretation of the binary specification: as a conditional mean specification. It follows that we can write the binary model as a regression model:

$$y_i = (1 - F(-x_i'\beta)) + \epsilon_i, \tag{21.8}$$

where ϵ_i is a residual representing the deviation of the binary y_i from its conditional mean. Then:

$$\begin{aligned} &\mathrm{E}(\epsilon_i | x_i, \beta) = 0 \\ &\mathrm{var}(\epsilon_i | x_i, \beta) = F(-x_i'\beta)(1 - F(-x_i'\beta)). \end{aligned} \tag{21.9}$$

We will use the conditional mean interpretation in our discussion of binary model residuals (see "Make Residual Series" on page 634).

Estimating Binary Models in EViews

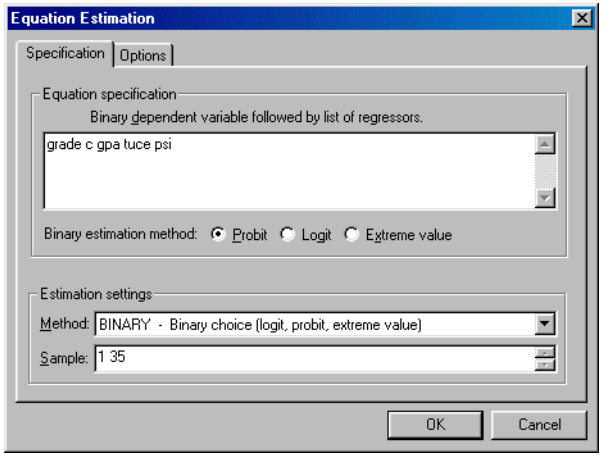
To estimate a binary dependent variable model, choose **Object/New Object...** from the main menu and select the **Equation** object from the main menu. From the Equation Specification dialog, select the **BINARY** estimation method. The dialog will change to reflect your choice.

There are two parts to the binary model specification. First, in the **Equation Specification** field, you should type the name of the binary dependent variable followed by a list of regressors. You may not enter an explicit equation since binary estimation only supports specification by list. Next, select from among the three distributions for your error term:

Probit	$\Pr(y_i = 1 \| x_i, \beta) = 1 - \Phi(-x_i'\beta) = \Phi(x_i'\beta)$ where Φ is the cumulative distribution function of the standard normal distribution.
Logit	$\Pr(y_i = 1 \| x_i, \beta) = 1 - (e^{-x_i'\beta}/(1 + e^{-x_i'\beta})) = e^{x_i'\beta}/(1 + e^{x_i'\beta})$ which is based upon the cumulative distribution function for the logistic distribution.

Extreme value (Gompit)	$\Pr(y_i = 1 \mid x_i, \beta) = 1 - (1 - \exp(-e^{-x_i'\beta})) = \exp(-e^{-x_i'\beta})$ which is based upon the CDF for the Type-I extreme value distribution. Note that this distribution is skewed.

For example, consider the probit specification example described in Greene (1997, p. 876) where we analyze the effectiveness of teaching methods on grades. The variable GRADE represents improvement on grades following exposure to the new teaching method PSI. Also controlling for alternative measures of knowledge (GPA and TUCE), we have the specification:



Once you have specified the model, click **OK**. EViews estimates the parameters of the model using iterative procedures, and will display information in the status line. EViews requires that the dependent variable be coded with the values zero-one with all other observations dropped from the estimation.

Following estimation, EViews displays results in the equation window. The top part of the estimation output is given by:

Dependent Variable: GRADE
Method: ML - Binary Probit
Date: 07/31/00 Time: 15:57
Sample: 1 32
Included observations: 32
Convergence achieved after 5 iterations
Covariance matrix computed using second derivatives

Variable	Coefficient	Std. Error	z-Statistic	Prob.
C	-7.452320	2.542472	-2.931131	0.0034
GPA	1.625810	0.693882	2.343063	0.0191
TUCE	0.051729	0.083890	0.616626	0.5375
PSI	1.426332	0.595038	2.397045	0.0165

The header contains basic information regarding the estimation technique (ML for maximum likelihood) and the sample used in estimation, as well as information on the number of iterations required for convergence, and on the method used to compute the coefficient covariance matrix.

Displayed next are the coefficient estimates, asymptotic standard errors, z-statistics and corresponding p-values.

Interpretation of the coefficient values is complicated by the fact that estimated coefficients from a binary model cannot be interpreted as the marginal effect on the dependent variable. The marginal effect of x_j on the conditional probability is given by:

$$\frac{\partial \mathrm{E}(y_i | x_i, \beta)}{\partial x_{ij}} = f(-x_i'\beta)\beta_j, \tag{21.10}$$

where $f(x) = dF(x)/dx$ is the density function corresponding to F. Note that β_j is weighted by a factor f that depends on the values of *all* of the regressors in x. The *direction* of the effect of a change in x_j depends only on the sign of the β_j coefficient. Positive values of β_j imply that increasing x_j will increase the probability of the response; negative values imply the opposite.

While marginal effects calculation is not provided as a built-in view or procedure, in "Forecast" on page 633, we show you how to use EViews to compute the marginal effects.

An alternative interpretation of the coefficients results from noting that the ratios of coefficients provide a measure of the relative changes in the probabilities:

$$\frac{\beta_j}{\beta_k} = \frac{\partial \mathrm{E}(y_i | x_i, \beta)/\partial x_{ij}}{\partial \mathrm{E}(y_i | x_i, \beta)/\partial x_{ik}}. \tag{21.11}$$

In addition to the summary statistics of the dependent variable, EViews also presents the following summary statistics:

Mean dependent var	0.343750	S.D. dependent var	0.482559
S.E. of regression	0.386128	Akaike info criterion	1.051175
Sum squared resid	4.174660	Schwarz criterion	1.234392
Log likelihood	-12.81880	Hannan-Quinn criter.	1.111907
Restr. log likelihood	-20.59173	Avg. log likelihood	-0.400588
LR statistic (3 df)	15.54585	McFadden R-squared	0.377478
Probability(LR stat)	0.001405		
Obs with Dep=0	21	Total obs	32
Obs with Dep=1	11		

First, there are several familiar summary descriptive statistics: the mean and standard deviation of the dependent variable, standard error of the regression, and the sum of the squared residuals. The latter two measures are computed in the usual fashion using the residuals:

$$e_i = y_i - \mathrm{E}(y_i | x_i, \hat{\beta}) = y_i - (1 - F(-x_i'\hat{\beta})). \tag{21.12}$$

Additionally, there are several likelihood based statistics:

- **Log likelihood** is the maximized value of the log likelihood function $l(\hat{\beta})$.
- **Avg. log likelihood** is the log likelihood $l(\hat{\beta})$ divided by the number of observations n.
- **Restr. log likelihood** is the maximized log likelihood value, when all slope coefficients are restricted to zero, $l(\tilde{\beta})$. Since the constant term is included, this specification is equivalent to estimating the unconditional mean probability of "success".
- The **LR statistic** tests the joint null hypothesis that all slope coefficients except the constant are zero and is computed as $-2(l(\tilde{\beta}) - l(\hat{\beta}))$. This statistic, which is only reported when you include a constant in your specification, is used to test the overall significance of the model. The number in parentheses is the degrees of freedom, which is the number of restrictions under test.
- **Probability(LR stat)** is the p-value of the LR test statistic. Under the null hypothesis, the LR test statistic is asymptotically distributed as a χ^2 variable, with degrees of freedom equal to the number of restrictions under test.
- **McFadden R-squared** is the likelihood ratio index computed as $1 - l(\hat{\beta})/l(\tilde{\beta})$, where $l(\tilde{\beta})$ is the restricted log likelihood. As the name suggests, this is an analog to the R^2 reported in linear regression models. It has the property that it always lies between zero and one.
- The various information criteria are detailed in Appendix E, "Information Criteria", beginning on page 971. For additional discussion, see Grasa (1989).

Estimation Options

The iteration limit and convergence criterion may be set in the usual fashion by clicking on the **Options** tab in the **Equation Estimation** dialog. In addition, there are options that are specific to binary models. These options are described below.

Robust Covariances

For binary dependent variable models, EViews allows you to estimate the standard errors using quasi-maximum likelihood (**Huber/White**) or generalized linear model (**GLM**) methods. See "Technical Notes" on page 667 for a discussion of these two methods.

Click on the **Options** tab to bring up the settings, check the **Robust Covariance** box and select one of the two methods. When you estimate the binary model using this option, the header in the equation output will indicate the method used to compute the coefficient covariance matrix.

Starting Values

As with other estimation procedures, EViews allows you to specify starting values. In the options menu, select one of the items from the combo box. You can use the default EViews values, or you can choose a fraction of those values, zero coefficients, or user supplied values. To employ the latter, enter the coefficients in the C coefficient vector, and select **User Supplied** in the combo box.

The EViews default values are selected using a sophisticated algorithm that is specialized for each type of binary model. Unless there is a good reason to choose otherwise, we recommend that you use the default values.

Estimation Algorithm

By default, EViews uses quadratic hill-climbing to obtain parameter estimates. This algorithm uses the matrix of analytic second derivatives of the log likelihood in forming iteration updates and in computing the estimated covariance matrix of the coefficients.

If you wish, you can employ a different estimation algorithm: Newton-Raphson also employs second derivatives (without the diagonal weighting); BHHH uses first derivatives to determine both iteration updates and the covariance matrix estimates (see Appendix C, "Estimation and Solution Options", on page 956). To employ one of these latter methods, click on **Options** in the Equation specification dialog box, and select the desired method.

Estimation Problems

In general, estimation of binary models is quite straightforward, and you should experience little difficulty in obtaining parameter estimates. There are a few situations, however, where you may experience problems.

First, you may get the error message "Dependent variable has no variance." This error means that there is no variation in the dependent variable (the variable is always one or zero for all valid observations). This error most often occurs when EViews excludes the entire sample of observations for which y takes values other than zero or one, leaving too few observations for estimation.

You should make certain to recode your data so that the binary indicators take the values zero and one. This requirement is not as restrictive at it may first seem, since the recoding may easily be done using auto-series. Suppose, for example, that you have data where y takes the values 1000 and 2000. You could then use the boolean auto-series, "`y=1000`", or perhaps, "`y<1500`", as your dependent variable.

Second, you may receive an error message of the form "[xxxx] perfectly predicts binary response [success/failure]", where xxxx is a sample condition. This error occurs when one of the regressors contains a separating value for which all of the observations with values below the threshold are associated with a single binary response, and all of the values above the threshold are associated with the alternative response. In this circumstance, the method of maximum likelihood breaks down.

For example, if all values of the explanatory variable $x > 0$ are associated with $y = 1$, then x is a perfect predictor of the dependent variable, and EViews will issue an error message and stop the estimation procedure.

The only solution to this problem is to remove the offending variable from your specification. Usually, the variable has been incorrectly entered in the model, as when a researcher includes a dummy variable that is identical to the dependent variable (for discussion, see Greene, 1997).

Thirdly, you may experience the error, "Non-positive likelihood value observed for observation [xxxx]." This error most commonly arises when the starting values for estimation are poor. The default EViews starting values should be adequate for most uses. You may wish to check the Options dialog to make certain that you are not using user specified starting values, or you may experiment with alternative user-specified values.

Lastly, the error message "Near-singular matrix" indicates that EViews was unable to invert the matrix required for iterative estimation. This will occur if the model is not identified. It may also occur if the current parameters are far from the true values. If you believe the latter to be the case, you may wish to experiment with starting values or the estimation algorithm. The BHHH and quadratic hill-climbing algorithms are less sensitive to this particular problem than is Newton-Raphson.

Views of Binary Equations

EViews provides a number of standard views and procedures for binary models. For example, you can easily perform Wald or likelihood ratio tests by selecting **View/Coefficient**

Tests, and then choosing the appropriate test. In addition, EViews allows you to examine and perform tests using the residuals from your model. The ordinary residuals used in most calculations are described above—additional residual types are defined below. Note that some care should be taken in interpreting test statistics that use these residuals since some of the underlying test assumptions may not be valid in the current setting.

There are a number of additional specialized views and procedures which allow you to examine the properties and performance of your estimated binary model.

Categorical Regressor Stats

This view displays descriptive statistics (mean and standard deviation) for each regressor. The descriptive statistics are computed for the whole sample, as well as the sample broken down by the value of the dependent variable y.

Expectation-Prediction (Classification) Table

This view displays 2×2 tables of correct and incorrect classification based on a user specified prediction rule, and on expected value calculations. Click on **View/Expectation-Prediction Table**. EViews opens a dialog prompting you to specify a prediction cutoff value, p, lying between zero and one. Each observation will be classified as having a predicted probability that lies above or below this cutoff.

After you enter the cutoff value and click on **OK**, EViews will display four (bordered) 2×2 tables in the equation window. Each table corresponds to a contingency table of the predicted response classified against the observed dependent variable. The top two tables and associated statistics depict the classification results based upon the specified cutoff value:

Dependent Variable: GRADE
Method: ML - Binary Probit
Date: 07/31/00 Time: 15:57
Sample: 1 32
Included observations: 32
Prediction Evaluation (success cutoff C = 0.5)

	Estimated Equation			Constant Probability		
	Dep=0	Dep=1	Total	Dep=0	Dep=1	Total
P(Dep=1)<=C	18	3	21	21	11	32
P(Dep=1)>C	3	8	11	0	0	0
Total	21	11	32	21	11	32
Correct	18	8	26	21	0	21
% Correct	85.71	72.73	81.25	100.00	0.00	65.62
% Incorrect	14.29	27.27	18.75	0.00	100.00	34.38
Total Gain*	-14.29	72.73	15.62			
Percent Gain**	NA	72.73	45.45			

In the left-hand table, we classify observations as having predicted probabilities $\hat{p}_i = 1 - F(-x_i'\hat{\beta})$ that are above or below the specified cutoff value (here set to the default of 0.5). In the upper right-hand table, we classify observations using $\bar{p}$, the sample

proportion of $y = 1$ observations. This probability, which is constant across individuals, is the value computed from estimating a model that includes only the intercept term, C.

"Correct" classifications are obtained when the predicted probability is less than or equal to the cutoff and the observed $y = 0$, or when the predicted probability is greater than the cutoff and the observed $y = 1$. In the example above, 18 of the Dep = 0 observations and 8 of the Dep = 1 observations are correctly classified by the estimated model.

It is worth noting that in the statistics literature, what we term the expectation-prediction table is sometimes referred to as the *classification table*. The fraction of $y = 1$ observations that are correctly predicted is termed the *sensitivity*, while the fraction of $y = 0$ observations that are correctly predicted is known as *specificity*. In EViews, these two values, expressed in percentage terms, are labeled "% Correct". Overall, the estimated model correctly predicts 81.25% of the observations (85.71% of the Dep = 0 and 72.73% of the Dep = 1 observations).

The gain in the number of correct predictions obtained in moving from the right table to the left table provides a measure of the predictive ability of your model. The gain measures are reported in both absolute percentage increases (**Total Gain**), and as a percentage of the incorrect classifications in the constant probability model (**Percent Gain**). In the example above, the restricted model predicts that all 21 individuals will have Dep = 0. This prediction is correct for the 21 $y = 0$ observations, but is incorrect for the 11 $y = 1$ observations.

The estimated model improves on the Dep = 1 predictions by 72.73 percentage points, but does more poorly on the Dep = 0 predictions (-14.29 percentage points). Overall, the estimated equation is 15.62 percentage points better at predicting responses than the constant probability model. This change represents a 45.45 percent improvement over the 65.62 percent correct prediction of the default model.

The bottom portion of the equation window contains analogous prediction results based upon expected value calculations:

	Estimated Equation			Constant Probability		
	Dep=0	Dep=1	Total	Dep=0	Dep=1	Total
E(# of Dep=0)	16.89	4.14	21.03	13.78	7.22	21.00
E(# of Dep=1)	4.11	6.86	10.97	7.22	3.78	11.00
Total	21.00	11.00	32.00	21.00	11.00	32.00
Correct	16.89	6.86	23.74	13.78	3.78	17.56
% Correct	80.42	62.32	74.20	65.62	34.38	54.88
% Incorrect	19.58	37.68	25.80	34.38	65.62	45.12
Total Gain*	14.80	27.95	19.32			
Percent Gain**	43.05	42.59	42.82			

In the left-hand table, we compute the expected number of $y = 0$ and $y = 1$ observations in the sample. For example, **E(# of Dep = 0)** is computed as:

$$\sum_i \Pr(y_i = 0 | x_i, \beta) = \sum_i F(-x_i'\hat{\beta}), \tag{21.13}$$

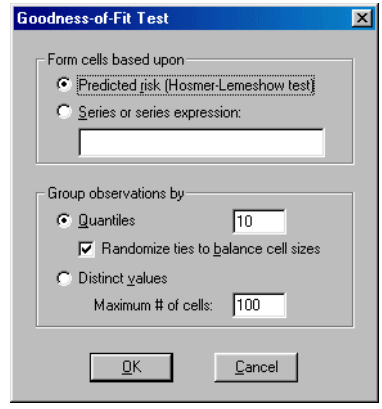
where the cumulative distribution function F is for the normal, logistic, or extreme value distribution.

In the lower right-hand table, we compute the expected number of $y = 0$ and $y = 1$ observations for a model estimated with only a constant. For this restricted model, **E(# of Dep = 0)** is computed as $n(1 - \bar{p})$, where $\bar{p}$ is the sample proportion of $y = 1$ observations. EViews also reports summary measures of the total gain and the percent (of the incorrect expectation) gain.

Among the 21 individuals with $y = 0$, the expected number of $y = 0$ observations in the estimated model is 16.89. Among the 11 observations with $y = 1$, the expected number of $y = 1$ observations is 6.86. These numbers represent roughly a 19.32 percentage point (42.82 percent) improvement over the constant probability model.

Goodness-of-Fit Tests

This view allows you to perform Pearson χ^2-type tests of goodness-of-fit. EViews carries out two goodness-of-fit tests: Hosmer-Lemeshow (1989) and Andrews (1988a, 1988b). The idea underlying these tests is to compare the fitted expected values to the actual values *by group*. If these differences are "large", we reject the model as providing an insufficient fit to the data.



Details on the two tests are described in the "Technical Notes" on page 667. Briefly, the tests differ in how the observations are grouped and in the asymptotic distribution of the test statistic. The Hosmer-Lemeshow test groups observations on the basis of the predicted probability that $y = 1$. The Andrews test is a more general test that groups observations on the basis of any series or series expression.

To carry out the test, select **View/Goodness-of-Fit Test…**

You must first decide on the grouping variable. You can select Hosmer-Lemeshow (predicted probability) grouping by clicking on the corresponding radio button, or you can select series grouping, and provide a series to be used in forming the groups.

Next, you need to specify the grouping rule. EViews allows you to group on the basis of either distinct values or quantiles of the grouping variable.

If your grouping variable takes relatively few distinct values, you should choose the **Distinct values** grouping. EViews will form a separate group for each distinct value of the grouping variable. For example, if your grouping variable is TUCE, EViews will create a group for each distinct TUCE value and compare the expected and actual numbers of $y = 1$ observations in each group. By default, EViews limits you to 100 distinct values. If the distinct values in your grouping series exceeds this value, EViews will return an error message. If you wish to evaluate the test for more than 100 values, you must explicitly increase the maximum number of distinct values.

If your grouping variable takes on a large number of distinct values, you should select **Quantiles**, and enter the number of desired bins in the edit field. If you select this method, EViews will group your observations into the number of specified bins, on the basis of the ordered values of the grouping series. For example, if you choose to group by TUCE, select **Quantiles**, and enter 10, EViews will form groups on the basis of TUCE deciles.

If you choose to group by quantiles and there are ties in the grouping variable, EViews may not be able to form the exact number of groups you specify unless tied values are assigned to different groups. Furthermore, the number of observations in each group may be very unbalanced. Selecting the **randomize ties** option randomly assigns ties to adjacent groups in order to balance the number of observations in each group.

Since the properties of the test statistics require that the number of observations in each group is "large", some care needs to be taken in selecting a rule so that you do not end up with a large number of cells, each containing small numbers of observations.

By default, EViews will perform the test using Hosmer-Lemeshow grouping. The default grouping method is to form deciles. The test result using the default specification is given by:

Dependent Variable: GRADE
Method: ML - Binary Probit
Date: 07/31/00 Time: 15:57
Sample: 1 32
Included observations: 32
Andrews and Hosmer-Lemeshow Goodness-of-Fit Tests
Grouping based upon predicted risk (randomize ties)

	Quantile of Risk		Dep=0		Dep=1		Total	H-L
	Low	High	Actual	Expect	Actual	Expect	Obs	Value
1	0.0161	0.0185	3	2.94722	0	0.05278	3	0.05372
2	0.0186	0.0272	3	2.93223	0	0.06777	3	0.06934
3	0.0309	0.0457	3	2.87888	0	0.12112	3	0.12621
4	0.0531	0.1088	3	2.77618	0	0.22382	3	0.24186
5	0.1235	0.1952	2	3.29779	2	0.70221	4	2.90924
6	0.2732	0.3287	3	2.07481	0	0.92519	3	1.33775
7	0.3563	0.5400	2	1.61497	1	1.38503	3	0.19883
8	0.5546	0.6424	1	1.20962	2	1.79038	3	0.06087
9	0.6572	0.8342	0	0.84550	3	2.15450	3	1.17730
10	0.8400	0.9522	1	0.45575	3	3.54425	4	0.73351
		Total	21	21.0330	11	10.9670	32	6.90863

H-L Statistic:	6.9086	Prob. Chi-Sq(8)	0.5465
Andrews Statistic:	20.6045	Prob. Chi-Sq(10)	0.0240

The columns labeled "Quantiles of Risk" depict the high and low value of the predicted probability for each decile. Also depicted are the actual and expected number of observations in each group, as well as the contribution of each group to the overall Hosmer-Lemeshow (H-L) statistic—large values indicate large differences between the actual and predicted values for that decile.

The χ^2 statistics are reported at the bottom of the table. Since grouping on the basis of the fitted values falls within the structure of an Andrews test, we report results for both the H-L and the Andrews test statistic. The p-value for the HL test is large while the value for the Andrews test statistic is small, providing mixed evidence of problems. Furthermore, the relatively small sample sizes suggest that caution is in order in interpreting the results.

Procedures for Binary Equations

In addition to the usual procedures for equations, EViews allows you to forecast the dependent variable and linear index, or to compute a variety of residuals associated with the binary model.

Forecast

EViews allows you to compute either the fitted probability, $\hat{p}_i = 1 - F(-x_i'\hat{\beta})$, or the fitted values of the index $x_i'\beta$. From the equation toolbar select **Proc/Forecast (Fitted Probability/Index)...**, and then click on the desired entry.

As with other estimators, you can select a forecast sample, and display a graph of the forecast. If your explanatory variables, x_t, include lagged values of the binary dependent vari-

able y_t, forecasting with the **Dynamic** option instructs EViews to use the fitted values $\hat{p}_{t-1}$, to derive the forecasts, in contrast with the **Static** option, which uses the actual (lagged) y_{t-1}.

Neither forecast evaluations nor automatic calculation of standard errors of the forecast are currently available for this estimation method. The latter can be computed using the variance matrix of the coefficients displayed by **View/Covariance Matrix**, or using the `@covariance` function.

You can use the fitted index in a variety of ways, for example, to compute the marginal effects of the explanatory variables. Simply forecast the fitted index and save the results in a series, say XB. Then the auto-series `@dnorm(-xb)`, `@dlogistic(-xb)`, or `@dextreme(-xb)` may be multiplied by the coefficients of interest to provide an estimate of the derivatives of the expected value of y_i with respect to the j-th variable in x_i:

$$\frac{\partial \mathrm{E}(y_i | x_i, \beta)}{\partial x_{ij}} = f(-x_i'\beta)\beta_j. \tag{21.14}$$

Make Residual Series

Proc/Make Residual Series gives you the option of generating one of the following three types of residuals:

Ordinary	$e_{oi} = y_i - \hat{p}_i$
Standardized	$e_{si} = \dfrac{y_i - \hat{p}_i}{\sqrt{\hat{p}_i(1-\hat{p}_i)}}$
Generalized	$e_{gi} = \dfrac{(y_i - \hat{p}_i) f(-x_i'\hat{\beta})}{\hat{p}_i(1-\hat{p}_i)}\phi$

where $\hat{p}_i = 1 - F(-x_i'\hat{\beta})$ is the fitted probability, and the distribution and density functions F and f, depend on the specified distribution.

The ordinary residuals have been described above. The standardized residuals are simply the ordinary residuals divided by an estimate of the theoretical standard deviation. The generalized residuals are derived from the first order conditions that define the ML estimates. The first order conditions may be regarded as an orthogonality condition between the generalized residuals and the regressors x.

$$\frac{\partial l(\beta)}{\partial \beta} = \sum_{i=1}^{N} \frac{(y_i - (1 - F(-x_i'\beta)))f(-x_i'\beta)}{F(-x_i'\beta)(1 - F(-x_i'\beta))} \cdot x_i = \sum_{i=1}^{N} e_{g,i} \cdot x_i. \tag{21.15}$$

This property is analogous to the orthogonality condition between the (ordinary) residuals and the regressors in linear regression models.

The usefulness of the generalized residuals derives from the fact that you can easily obtain the score vectors by multiplying the generalized residuals by each of the regressors in x. These scores can be used in a variety of LM specification tests (see Chesher, Lancaster and Irish (1985), and Gourieroux, Monfort, Renault, and Trognon (1987)). We provide an example below.

Demonstrations

You can easily use the results of a binary model in additional analysis. Here, we provide demonstrations of using EViews to plot a probability response curve and to test for heteroskedasticity in the residuals.

Plotting Probability Response Curves

You can use the estimated coefficients from a binary model to examine how the predicted probabilities vary with an independent variable. To do so, we will use the EViews built-in modeling features.

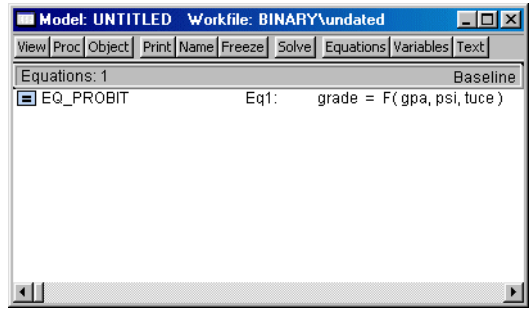
For the probit example above, suppose we are interested in the effect of teaching method (PSI) on educational improvement (GRADE). We wish to plot the fitted probabilities of GRADE improvement as a function of GPA for the two values of PSI, fixing the values of other variables at their sample means.

We will perform the analysis using a grid of values for GPA from 2 to 4. First, we will create a series containing the values of GPA for which we wish to examine the fitted probabilities for GRADE. The easiest way to do this is to use the `@trend` function to generate a new series:

```
series gpa_plot=2+(4-2)*@trend/(@obs(@trend)-1)
```

`@trend` creates a series that begins at 0 in the first observation of the sample, and increases by 1 for each subsequent observation, up through @obs-1.

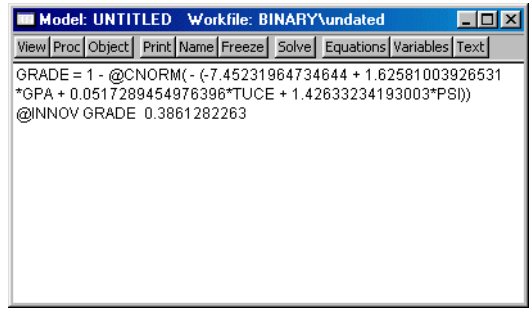
Next, we will use a model object to define and perform the desired computations. The following discussion skims over many of the useful features of EViews models. Those wishing greater detail should consult Chapter 26, “Models”, beginning on page 777.



First, we create a model out of the estimated equation by selecting **Proc/Make Model** from the equation toolbar. EViews will create an untitled model object containing a link to the estimated equation and will open the model window.

Next we want to edit this model specification so that calculations are performed using our simulation values. To do so we must first break the link between the original equation and the model specification by selecting **Proc/Links/Break All Links**. Next, click on the Text button or select **View/Source Text** to display the text editing screen.

We wish to create two separate equations: one with the value of PSI set to 0 and one with the value of PSI set to 1 (you can, of course, use copy-and-paste to aid in creating the additional equation). We will also edit the specification so that references to GPA are replaced with the series of simulation values GPA_PLOT, and references to TUCE are replaced with the calculated mean, "@MEAN(TUCE)". The GRADE_0 equation sets PSI to 0, while the GRADE_1 contains an additional expression, 1.426332342, which is the coefficient on the PSI variable.


Model: UNTITLED Workfile: BINARY\undated
View Proc Object Print Name Freeze Solve Equations Variables Text
GRADE = 1 - @CNORM(- (-7.45231964734644 + 1.62581003926531
*GPA + 0.0517289454976396*TUCE + 1.42633234193003*PSI))
@INNOV GRADE 0.3861282263

Once you have edited your model, click on **Solve** and set the "Active" solution scenario to "Actuals". This tells EViews that you wish to place the solutions in the series "GRADE_0" and "GRADE_1" as specified in the equation definitions. You can safely ignore the remaining solution settings and simply click on **OK**. EViews will report that your model has solved successfully.

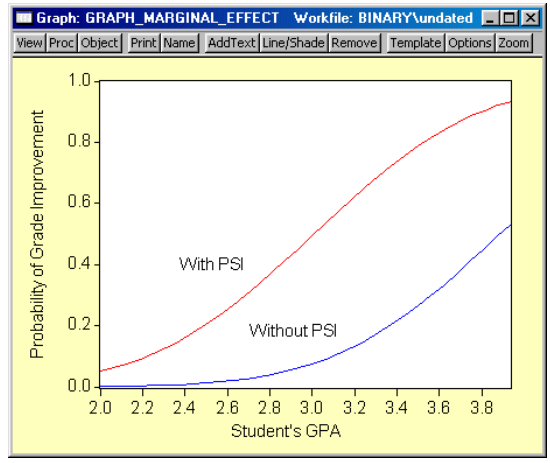
You are now ready to plot results. Select **Object/New Object.../Group**, and enter:

```
gpa_plot grade_0 grade_1
```

EViews will open an untitled group window containing these two series. Select **View/Graph/XY line** to display the probability of GRADE improvement plotted against GPA for those with and without PSI (and with the TUCE evaluated at means).

EViews will open a group window containing these series. Select **View/Graph/XY line** from the group toolbar to display a graph of your results.

We have annotated the graph slightly so that you can better judge the effect of the new teaching methods (PSI) on the GPA—Grade Improvement relationship.



Testing for Heteroskedasticity

As an example of specification tests for binary dependent variable models, we carry out the LM test for heteroskedasticity using the artificial regression method described by Davidson and MacKinnon (1993, section 15.4). We test the null hypothesis of homoskedasticity against the alternative of heteroskedasticity of the form:

$$\text{var}(u_i) = \exp(2z_i'\gamma), \tag{21.16}$$

where γ is an unknown parameter. In this example, we take PSI as the only variable in z. The test statistic is the explained sum of squares from the regression:

$$\frac{(y_i - \hat{p}_i)}{\sqrt{\hat{p}_i(1-\hat{p}_i)}} = \frac{f(-x_i'\hat{\beta})}{\sqrt{\hat{p}_i(1-\hat{p}_i)}} x_i' b_1 + \frac{f(-x_i'\hat{\beta})(-x_i'\hat{\beta})}{\sqrt{\hat{p}_i(1-\hat{p}_i)}} z_i' b_2 + v_i, \tag{21.17}$$

which is asymptotically distributed as a χ^2 with degrees of freedom equal to the number of variables in z (in this case 1).

To carry out the test, we first retrieve the fitted probabilities $\hat{p}_i$ and fitted index $x_i'\beta$. Click on the **Forecast** button and first save the fitted probabilities as P_HAT and then the index as XB (you will have to click **Forecast** twice to save the two series).

Next, the dependent variable in the test regression may be obtained as the standardized residual. Select **Proc/Make Residual Series...** and select **Standardized Residual**. We will save the series as BRMR_Y.

Lastly, we will use the built-in EViews functions for evaluating the normal density and cumulative distribution function to create a group object containing the independent variables:

```
series fac=@dnorm(-xb)/@sqrt(p_hat*(1-p_hat))
group brmr_x fac (gpa*fac) (tuce*fac) (psi*fac)
```

Then run the artificial regression by clicking on **Quick/Estimate Equation…,** selecting **Least Squares,** and entering:

```
brmr_y brmr_x (psi*(-xb)*fac)
```

You can obtain the fitted values by clicking on the **Forecast** button in the equation toolbar of this artificial regression. The LM test statistic is the sum of squares of these fitted values. If the fitted values from the artificial regression are saved in BRMR_YF, the test statistic can be saved as a scalar named LM_TEST:

```
scalar lm_test=@sumsq(brmr_yf)
```

which contains the value 1.5408. You can compare the value of this test statistic with the critical values from the chi-square table with one degree of freedom. To save the p-value as a scalar, enter the command:

```
scalar p_val=1-@cchisq(lm_test,1)
```

To examine the value of LM_TEST or P_VAL, double click on the name in the workfile window; the value will be displayed in the status line at the bottom of the EViews window. The p-value in this example is roughly 0.21, so we have little evidence against the null hypothesis of homoskedasticity.

Ordered Dependent Variable Models

EViews estimates the ordered-response model of Aitchison and Silvey (1957) under a variety of assumptions about the latent error distribution. In ordered dependent variable models, the observed y denotes outcomes representing ordered or ranked categories. For example, we may observe individuals who choose between one of four educational outcomes: less than high school, high school, college, advanced degree. Or we may observe individuals who are employed, partially retired, or fully retired.

As in the binary dependent variable model, we can model the observed response by considering a latent variable y_i^* that depends linearly on the explanatory variables x_i:

$$y_i^* = x_i'\beta + \epsilon_i \tag{21.18}$$

where is ϵ_i are independent and identically distributed random variables. The observed y_i is determined from y_i^* using the rule:

$$y_i = \begin{cases} 0 & \text{if } y_i^* \le \gamma_1 \\ 1 & \text{if } \gamma_1 < y_i^* \le \gamma_2 \\ 2 & \text{if } \gamma_2 < y_i^* \le \gamma_2 \\ \vdots & \vdots \\ M & \text{if } \gamma_M < y_i^* \end{cases} \tag{21.19}$$

It is worth noting that the actual values chosen to represent the categories in y are completely arbitrary. All the ordered specification requires is for ordering to be preserved so that $y_i^* < y_j^*$ implies that $y_i < y_j$.

It follows that the probabilities of observing each value of y are given by

$$\begin{aligned} \Pr(y_i = 0|x_i, \beta, \gamma) &= F(\gamma_1 - x_i'\beta) \\ \Pr(y_i = 1|x_i, \beta, \gamma) &= F(\gamma_2 - x_i'\beta) - F(\gamma_1 - x_i'\beta) \\ \Pr(y_i = 2|x_i, \beta, \gamma) &= F(\gamma_3 - x_i'\beta) - F(\gamma_2 - x_i'\beta) \\ &\cdots \\ \Pr(y_i = M|x_i, \beta, \gamma) &= 1 - F(\gamma_M - x_i'\beta) \end{aligned} \tag{21.20}$$

where F is the cumulative distribution function of ϵ.

The threshold values γ are estimated along with the β coefficients by maximizing the log likelihood function:

$$l(\beta, \gamma) = \sum_{i=1}^{N} \sum_{j=0}^{M} \log(\Pr(y_i = j|x_i, \beta, \gamma)) \cdot 1(y_i = j) \tag{21.21}$$

where $1(.)$ is an indicator function which takes the value 1 if the argument is true, and 0 if the argument is false. By default, EViews uses analytic second derivative methods to obtain parameter and variance matrix of the estimated coefficient estimates (see "Quadratic hill-climbing (Goldfeld-Quandt)" on page 957).

Estimating Ordered Models in EViews

Suppose that the dependent variable DANGER is an index ordered from 1 (least dangerous animal) to 5 (most dangerous animal). We wish to model this ordered dependent variable as a function of the explanatory variables, BODY, BRAIN and SLEEP. Note that the values that we have assigned to the dependent variable are not relevant, only the ordering implied by those values. EViews will estimate an identical model if the dependent variable is recorded to take the values 1, 2, 3, 4, 5 or 10, 234, 3243, 54321, 123456.

To estimate this model, select **Quick/Estimate Equation…** from the main menu. From the **Equation Estimation** dialog, select estimation method **ORDERED**. The standard estimation dialog will change to match this specification.

There are three parts to specifying an ordered variable model: the equation specification, the error specification, and the sample specification. First, in the **Equation specification** field, you should type the name of the ordered dependent variable followed by the list of your regressors. In our example, you will enter:

```
danger body brain sleep
```

Ordered estimation only supports specification by list so you may not enter an explicit equation.

Also keep in mind that:

- A separate constant term is not separately identified from the limit points γ, so EViews will ignore any constant term in your specification. Thus, the model:

  ```
  danger c body brain sleep
  ```

  is equivalent to the specification above.

- EViews requires the dependent variable to be integer valued, otherwise you will see an error message, and estimation will stop. This is not, however, a serious restriction, since you can easily convert the series into an integer using `@round`, `@floor` or `@ceil` in an auto-series expression.

Next, select between the ordered logit, ordered probit, and the ordered extreme value models by choosing one of the three distributions for the latent error term.

Lastly, specify the estimation sample.

Now click on **OK**, EViews will estimate the parameters of the model using iterative procedures.

Once the estimation procedure converges, EViews will display the estimation results in the equation window. The first part of the table contains the usual header information, including the assumed error distribution, estimation sample, iteration and convergence information, number of distinct values for y, and the method of computing the coefficient covariance matrix.

Dependent Variable: DANGER
Method: ML - Ordered Probit
Date: 09/13/97 Time: 10:00
Sample(adjusted): 1 61
Included observations: 58
Excluded observations: 3 after adjusting endpoints
Number of ordered indicator values: 5
Convergence achieved after 5 iterations
Covariance matrix computed using second derivatives

	Coefficient	Std. Error	z-Statistic	Prob.
BODY	0.006346	0.003262	1.945385	0.0517
BRAIN	-0.003506	0.001822	-1.924244	0.0543
SLEEP	-0.158596	0.040440	-3.921741	0.0001

Below the header information are the coefficient estimates and asymptotic standard errors, and the corresponding z-statistics and significance levels. The estimated coefficients of the ordered model must be interpreted with care (see Greene (1997, section 19.8) or Johnston and DiNardo (1997, section 13.9)).

The *sign* of $\hat{\beta}_j$ shows the direction of the change in the probability of falling in the end-point rankings ($y = 0$ or $y = 1$) when x_{ij} changes. Pr($y = 0$) changes in the *opposite* direction of the sign of β_j and Pr($y = M$) changes in the *same* direction as the sign of $\hat{\beta}_j$. The effects on the probability of falling in any of the middle rankings are given by:

$$\frac{\partial \Pr(y = k)}{\partial \beta_j} = \frac{\partial F(\gamma_{k+1} - x_i'\beta)}{\partial \beta_j} - \frac{\partial F(\gamma_k - x_i'\beta)}{\partial \beta_j} \tag{21.22}$$

for $k = 1, 2, \ldots, M-1$. It is impossible to determine the signs of these terms, *a priori*.

The lower part of the estimation output, labeled "Limit Points", presents the estimates of the γ coefficients and the associated standard errors and probability values:

LimitPoints				
Limit_2:C(4)	-2.382697	0.512993	-4.644695	0.0000
Limit_3:C(5)	-1.598777	0.484884	-3.297237	0.0010
Limit_4:C(6)	-1.028655	0.465433	-2.210104	0.0271
Limit_5:C(7)	-0.241152	0.445500	-0.541307	0.5883
Akaike info criterion	11.74480	Schwarz criterion		11.99348
Log likelihood	-333.5993	Hannan-Quinn criter.		11.84167
Avg. log likelihood	-5.751712			

Note that the coefficients are labeled both with the identity of the limit point, and the coefficient number. Just below the limit points are the summary statistics for the equation.

Estimation Problems

Most of the previous discussion of estimation problems for binary models (page 627) also holds for ordered models. In general, these models are well-behaved and will require little intervention.

There are cases, however, where problems will arise. First, EViews currently has a limit of 750 total coefficients in an ordered dependent variable model. Thus, if you have 25 right-hand side variables, and a dependent variable with 726 distinct values, you will be unable to estimate your model using EViews.

Second, you may run into identification problems and estimation difficulties if you have some groups where there are very few observations. If necessary, you may choose to combine adjacent groups and re-estimate the model.

EViews may stop estimation with the message "Parameter estimates for limit points are non-ascending", most likely on the first iteration. This error indicates that parameter values for the limit points were invalid, and that EViews was unable to adjust these values to make them valid. Make certain that if you are using user defined parameters, the limit points are strictly increasing. Better yet, we recommend that you employ the EViews starting values since they are based on a consistent first-stage estimation procedure, and should therefore be quite well-behaved.

Views of Ordered Equations

EViews provides you with several views of an ordered equation. As with other equations, you can examine the specification and estimated covariance matrix as well as perform Wald and likelihood ratio tests on coefficients of the model. In addition, there are several views that are specialized for the ordered model:

- **Dependent Variable Frequencies** — computes a one-way frequency table for the ordered dependent variable for the observations in the estimation sample. EViews presents both the frequency table and the cumulative frequency table in levels and percentages.
- **Expectation-Prediction Table** — classifies observations on the basis of the predicted response. EViews performs the classification on the basis of maximum predicted probability as well as the expected probability.

Dependent Variable: DANGER
Method: ML - Ordered Probit
Date: 09/13/97 Time: 10:00
Sample(adjusted): 1 61
Included observations: 58
Excluded observations: 3 after adjusting endpoints
Prediction table for ordered dependent variable

Value	Count	Count of obs with Max Prob	Error	Sum of all Probabilities	Error
1	18	27	-9	18.571	-0.571
2	14	16	-2	13.417	0.583
3	10	0	10	9.163	0.837
4	9	8	1	8.940	0.060
5	7	7	0	7.909	-0.909

There are two columns labeled "Error". The first measures the difference between the observed count and the number of observations where the probability of that response is highest. For example, 18 individuals reported a value of 1 for DANGER, while 27 individuals had predicted probabilities that were highest for this value. The actual count minus the predicted is –9. The second error column measures the difference between the actual number of individuals reporting the value, and the sum of all of the individual probabilities for that value.

Procedures for Ordered Equations

Make Ordered Limit Vector/Matrix

The full set of coefficients and the covariance matrix may be obtained from the estimated equation in the usual fashion (see "Working With Equation Statistics" on page 454). In some circumstances, however, you may wish to perform inference using only the estimates of the γ coefficients and the associated covariances.

The **Make Ordered Limit Vector** and **Make Ordered Limit Covariance Matrix** procedures provide a shortcut method of obtaining the estimates associated with the γ coefficients. The first procedure creates a vector (using the next unused name of the form LIMITS01, LIMITS02, etc.) containing the estimated γ coefficients. The latter procedure creates a symmetric matrix containing the estimated covariance matrix of the γ. The matrix will be given an unused name of the form VLIMITS01, VLIMITS02, etc., where the "V" is used to indicate that these are the variances of the estimated limit points.

Forecasting using Models

You cannot forecast directly from an estimated ordered model since the dependent variable represents categorical or rank data. EViews does, however, allow you to forecast the probability associated with each category. To forecast these probabilities, you must first create a

model. Choose **Proc/Make Model** and EViews will open an untitled model window containing a system of equations, with a separate equation for the probability of each ordered response value.

To forecast from this model, simply click the Solve button in the model window toolbar. If you select Scenario 1 as your solution scenario, the default settings will save your results in a set of named series with "_1" appended to the end of the each underlying name. See Chapter 26, "Models", beginning on page 777 for additional detail on modifying and solving models.

For this example, the series I_DANGER_1 will contain the fitted linear index $x_i'\hat{\beta}$. The fitted probability of falling in category 1 will be stored as a series named DANGER_1_1, the fitted probability of falling in category 2 will be stored as a series named DANGER_2_1, and so on. Note that for each observation, the fitted probability of falling in each of the categories sums up to one.

Make Residual Series

The generalized residuals of the ordered model are the derivatives of the log likelihood with respect to a hypothetical unit- x variable. These residuals are defined to be uncorrelated with the explanatory variables of the model (see Chesher and Irish (1987), and Gourieroux, Monfort, Renault and Trognon (1987) for details), and thus may be used in a variety of specification tests.

To create a series containing the generalized residuals, select **View/Make Residual Series...**, enter a name or accept the default name, and click **OK**. The generalized residuals for an ordered model are given by:

$$e_{gi} = \frac{f(\gamma_g - x_i'\hat{\beta}) - f(\gamma_{g-1} - x_i'\hat{\beta})}{F(\gamma_g - x_i'\hat{\beta}) - F(\gamma_{g-1} - x_i'\hat{\beta})}, \tag{21.23}$$

where $\gamma_0 = -\infty$, and $\gamma_{M+1} = \infty$.

Censored Regression Models

In some settings, the dependent variable is only partially observed. For example, in survey data, data on incomes above a specified level are often top-coded to protect confidentiality. Similarly desired consumption on durable goods may be censored at a small positive or zero value. EViews provides tools to perform maximum likelihood estimation of these models and to use the results for further analysis.

Theory

Consider the following latent variable regression model:

$$y_i^* = x_i'\beta + \sigma\epsilon_i, \tag{21.24}$$

where σ is a scale parameter. The scale parameter σ is identified in censored and truncated regression models, and will be estimated along with the β.

In the canonical *censored regression model*, known as the *tobit* (when there are normally distributed errors), the observed data y are given by:

$$y_i = \begin{cases} 0 & \text{if } y_i^* \le 0 \\ y_i^* & \text{if } y_i^* > 0 \end{cases} \tag{21.25}$$

In other words, all negative values of y_i^* are coded as 0. We say that these data are *left censored* at 0. Note that this situation differs from a *truncated regression model* where negative values of y_i^* are dropped from the sample. More generally, EViews allows for both left and right censoring at arbitrary limit points so that:

$$y_i = \begin{cases} \underline{c}_i & \text{if } y_i^* \le \underline{c}_i \\ y_i^* & \text{if } \underline{c}_i < y_i^* \le \bar{c}_i \\ \bar{c}_i & \text{if } \bar{c}_i < y_i^* \end{cases} \tag{21.26}$$

where $\underline{c}_i$, $\bar{c}_i$ are fixed numbers representing the censoring points. If there is no left censoring, then we can set $\underline{c}_i = -\infty$. If there is no right censoring, then $\bar{c}_i = \infty$. The canonical tobit model is a special case with $\underline{c}_i = 0$ and $\bar{c}_i = \infty$.

The parameters β, σ are estimated by maximizing the log likelihood function:

$$\begin{aligned} l(\beta, \sigma) = \sum_{i=1}^{N} & \log f((y_i - x_i'\beta)/\sigma) \cdot 1(\underline{c}_i < y_i < \bar{c}_i) \\ & + \log(F((\underline{c}_i - x_i'\beta)/\sigma)) \cdot 1(y_i = \underline{c}_i) \\ & + \log(1 - F((\underline{c}_i - x_i'\beta)/\sigma)) \cdot 1(y_i = \bar{c}_i) \end{aligned} \tag{21.27}$$

where f, F are the density and cumulative distribution functions of ϵ, respectively.

Estimating Censored Models in EViews

Consider the model:

$$\text{HRS}_i = \beta_1 + \beta_2 \text{AGE}_i + \beta_3 \text{EDU}_i + \beta_4 \text{KID1}_i + \epsilon_i, \tag{21.28}$$

where hours worked (HRS) is left censored at zero. To estimate this model, select **Quick/ Estimate Equation…** from the main menu. Then from the **Equation Estimation** dialog,

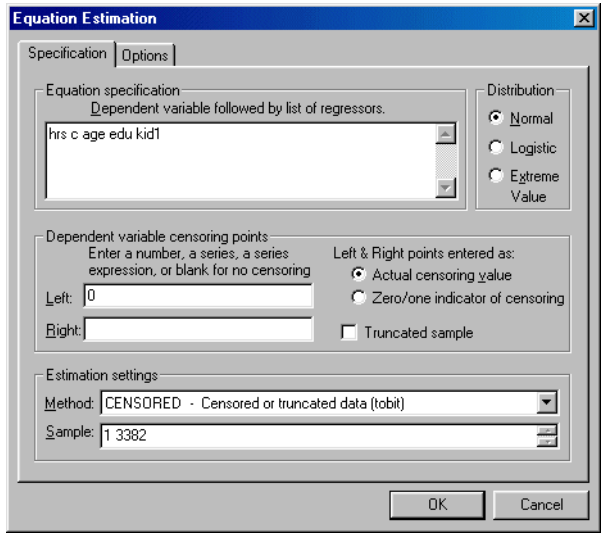
select the **CENSORED** estimation method. The dialog will change to provide a number of different input options.

Specifying the Regression Equation

In the **Equation specification** field, enter the name of the censored dependent variable followed by a list of regressors. In our example, you will enter:

```
hrs c age edu kid1
```

Censored estimation only supports specification by list so you may not enter an explicit equation.



Next, select one of the three distributions for the error term. EViews allows you three possible choices for the distribution of ϵ:

Standard normal	$\mathrm{E}(\epsilon) = 0$, $\mathrm{var}(\epsilon) = 1$
Logistic	$\mathrm{E}(\epsilon) = 0$, $\mathrm{var}(\epsilon) = \pi^2/3$
Extreme value (Type I)	$\mathrm{E}(\epsilon) \approx 0.5772$ (Euler's constant), $\mathrm{var}(\epsilon) = \pi^2/6$

Specifying the Censoring Points

You must also provide information about the censoring points of the dependent variable. There are two cases to consider: (1) where the limit points are known for all individuals,

and (2) where the censoring is by indicator and the limit points are known only for individuals with censored observations.

Limit Points Known

You should enter expressions for the left and right censoring points in the edit fields as required. Note that if you leave an edit field blank, EViews will assume that there is no censoring of observations of that type.

For example, in the canonical tobit model the data are censored on the left at zero, and are uncensored on the right. This case may be specified as:

Left edit field: `0`

Right edit field: *[blank]*

Similarly, top-coded censored data may be specified as,

Left edit field: *[blank]*

Right edit field: `20000`

while the more general case of left and right censoring is given by:

Left edit field: `10000`

Right edit field: `20000`

EViews also allows more general specifications where the censoring points are known to differ across observations. Simply enter the name of the series or auto-series containing the censoring points in the appropriate edit field. For example:

Left edit field: `lowinc`

Right edit field: `vcens1+10`

specifies a model with LOWINC censoring on the left-hand side, and right censoring at the value of VCENS1 + 10.

Limit Points Not Known

In some cases, the hypothetical censoring point is unknown for some individuals ($\underline{c}_i$ and $\bar{c}_i$ are not observed for all observations). This situation often occurs with data where censoring is indicated with a zero-one dummy variable, but no additional information is provided about potential censoring points.

EViews provides you an alternative method of describing data censoring that matches this format. Simply select the **Field is zero/one indicator of censoring** option in the estimation dialog, and enter the series expression for the censoring indicator(s) in the appropriate edit field(s). Observations with a censoring indicator of one are assumed to be censored while those with a value of zero are assumed to be actual responses.

For example, suppose that we have observations on the length of time that an individual has been unemployed (U), but that some of these observations represent ongoing unemployment at the time the sample is taken. These latter observations may be treated as right censored at the reported value. If the variable RCENS is a dummy variable representing censoring, you can click on the **Field is zero/one indicator of censoring** setting and enter:

Left edit field: *[blank]*

Right edit field: `rcens`

in the edit fields. If the data are censored on both the left and the right, use separate binary indicators for each form of censoring:

Left edit field: `lcens`

Right edit field: `rcens`

where LCENS is also a binary indicator.

Once you have specified the model, click **OK**. EViews will estimate the parameters of the model using appropriate iterative techniques.

A Comparison of Censoring Methods

An alternative to specifying index censoring is to enter a very large positive or negative value for the censoring limit for non-censored observations. For example, you could enter "1e-100" and "1e100" as the censoring limits for an observation on a completed unemployment spell. In fact, any limit point that is "outside" the observed data will suffice.

While this latter approach will yield the same likelihood function and therefore the same parameter values and coefficient covariance matrix, there is a drawback to the artificial limit approach. The presence of a censoring value implies that it is possible to evaluate the conditional mean of the observed dependent variable, as well as the ordinary and standardized residuals. All of the calculations that use residuals will, however, be based upon the arbitrary artificial data and will be invalid.

If you specify your censoring by index, you are informing EViews that you do not have information about the censoring for those observations that are not censored. Similarly, if an observation is left censored, you may not have information about the right censoring limit. In these circumstances, you should specify your censoring by index so that EViews will prevent you from computing the conditional mean of the dependent variable and the associated residuals.

Interpreting the Output

If your model converges, EViews will display the estimation results in the equation window. The first part of the table presents the usual header information, including informa-

tion about the assumed error distribution, estimation sample, estimation algorithms, and number of iterations required for convergence.

EViews also provides information about the specification for the censoring. If the estimated model is the canonical tobit with left-censoring at zero, EViews will label the method as a TOBIT. For all other censoring methods, EViews will display detailed information about form of the left and/or right censoring.

Here, we have the header output from a left censored model where the censoring is specified by value:

```
Dependent Variable: Y_PT
Method: ML - Censored Normal (TOBIT)
Date: 09/14/97   Time: 08:27
Sample: 1 601
Included observations: 601
Convergence achieved after 8 iterations
Covariance matrix computed using second
derivatives
```

Below the header are the usual results for the coefficients, including the asymptotic standard errors, z-statistics, and significance levels. As in other limited dependent variable models, the estimated coefficients do not have a direct interpretation as the marginal effect of the associated regressor j for individual i, x_{ij}. In censored regression models, a change in x_{ij} has two effects: an effect on the mean of y, given that it is observed, and an effect on the probability of y being observed (see McDonald and Moffitt, 1980).

In addition to results for the regression coefficients, EViews reports an additional coefficient named SCALE, which is the estimated scale factor σ. This scale factor may be used to estimate the standard deviation of the residual, using the known variance of the assumed distribution. For example, if the estimated SCALE has a value of 0.446 for a model with extreme value errors, the implied standard error of the error term is $0.5977 = 0.466\pi/\sqrt{6}$.

Most of the other output is self-explanatory. As in the binary and ordered models above, EViews reports summary statistics for the dependent variable and likelihood based statistics. The regression statistics at the bottom of the table are computed in the usual fashion, using the residuals $\hat{\epsilon}_i = y_i - \mathrm{E}(y_i|x_i, \hat{\beta}, \hat{\sigma})$ from the observed y.

Views of Censored Equations

Most of the views that are available for a censored regression are familiar from other settings. The residuals used in the calculations are defined below.

The one new view is the **Categorical Regressor Stats** view, which presents means and standard deviations for the dependent and independent variables for the estimation sample. EViews provides statistics computed over the entire sample, as well as for the left censored, right censored and non-censored individuals.

Procedures for Censored Equations

EViews provides several procedures which provide access to information derived from your censored equation estimates.

Make Residual Series

Select **Proc/Make Residual Series**, and select from among the three types of residuals. The three types of residuals for censored models are defined as:

Ordinary	$e_{oi} = y_i - \mathrm{E}(y_i \mid x_i, \hat{\beta}, \hat{\sigma})$
Standardized	$e_{si} = \dfrac{y_i - \mathrm{E}(y_i \mid x_i, \hat{\beta}, \hat{\sigma})}{\sqrt{\mathrm{var}(y_i \mid x_i, \hat{\beta}, \hat{\sigma})}}$
Generalized	$e_{gi} = -\dfrac{f((\underline{c}_i - x_i'\hat{\beta})/\hat{\sigma})}{\sigma F((\underline{c}_i - x_i'\hat{\beta})/\hat{\sigma})} \cdot 1(y_i \le \underline{c}_i)$ $-\dfrac{f'((\underline{c}_i - x_i'\hat{\beta})/\hat{\sigma})}{\sigma F((\underline{c}_i - x_i'\hat{\beta})/\hat{\sigma})} \cdot 1(\underline{c}_i < y_i \le \bar{c}_i)$ $+\dfrac{f((\bar{c}_i - x_i'\hat{\beta})/\hat{\sigma})}{\sigma(1-F((\bar{c}_i - x_i'\hat{\beta})/\hat{\sigma}))} \cdot 1(y_i \le \bar{c}_i)$

where f, F are the density and distribution functions, and where 1 is an indicator function which takes the value 1 if the condition in parentheses is true, and 0 if it is false. All of the above terms will be evaluated at the estimated β and σ. See the discussion of forecasting for details on the computation of $\mathrm{E}(y_i \mid x_i, \beta, \sigma)$.

The generalized residuals may be used as the basis of a number of LM tests, including LM tests of normality (see Lancaster, Chesher and Irish (1985), Chesher and Irish (1987), and Gourioux, Monfort, Renault and Trognon (1987); Greene (1997), provides a brief discussion and additional references).

Forecasting

EViews provides you with the option of forecasting the expected dependent variable, $\mathrm{E}(y_i \mid x_i, \beta, \sigma)$, or the expected latent variable, $\mathrm{E}(y_i^* \mid x_i, \beta, \sigma)$. Select **Forecast** from the equation toolbar to open the forecast dialog.

To forecast the expected *latent variable*, click on **Index - Expected latent variable,** and enter a name for the series to hold the output. The forecasts of the expected latent variable $\mathrm{E}(y_i^* \mid x_i, \beta, \sigma)$ may be derived from the latent model using the relationship:

$$\hat{y}_i^* = \mathrm{E}(y_i^* \big| x_i, \hat{\beta}, \hat{\sigma}) = x_i'\hat{\beta}. \tag{21.29}$$

To forecast the expected *observed dependent variable*, you should select **Expected dependent variable**, and enter a series name. These forecasts are computed using the relationship:

$$\begin{aligned}\hat{y}_i = \mathrm{E}(y_i \big| x_i, \hat{\beta}, \hat{\sigma}) &= \underline{c}_i \cdot \Pr(y_i = \underline{c}_i \big| x_i, \hat{\beta}, \hat{\sigma}) \\ &+ \mathrm{E}(y_i^* \big| \underline{c}_i < y_i^* < \bar{c}_i;\ x_i, \hat{\beta}, \hat{\sigma}) \cdot \Pr(\underline{c}_i < y_i^* < \bar{c}_i \big| x_i, \hat{\beta}, \hat{\sigma}) \\ &+ \bar{c}_i \cdot \Pr(y_i = \underline{c}_i \big| x_i, \hat{\beta}, \hat{\sigma})\end{aligned} \tag{21.30}$$

Note that these forecasts always satisfy $\underline{c}_i \le \hat{y}_i \le \bar{c}_i$. The probabilities associated with being in the various classifications are computed by evaluating the cumulative distribution function of the specified distribution. For example, the probability of being at the lower limit is given by:

$$\Pr(y_i = \underline{c}_i \big| x_i, \hat{\beta}, \hat{\sigma}) = \Pr(y_i^* \le \underline{c}_i \big| x_i, \hat{\beta}, \hat{\sigma}) = F((\underline{c}_i - x_i'\hat{\beta})/\hat{\sigma}). \tag{21.31}$$

Censored Model Illustration

As an example, we replicate Fair's (1978) tobit model that estimates the incidence of extramarital affairs. The dependent variable, number of extramarital affairs (Y_PT), is left censored at zero and the errors are assumed to be normally distributed. The bottom portion of the output is presented below:

	Coefficient	Std. Error	z-Statistic	Prob.
C	7.608487	3.905837	1.947979	0.0514
Z1	0.945787	1.062824	0.889881	0.3735
Z2	-0.192698	0.080965	-2.380015	0.0173
Z3	0.533190	0.146602	3.636997	0.0003
Z4	1.019182	1.279524	0.796532	0.4257
Z5	-1.699000	0.405467	-4.190231	0.0000
Z6	0.025361	0.227658	0.111399	0.9113
Z7	0.212983	0.321145	0.663198	0.5072
Z8	-2.273284	0.415389	-5.472657	0.0000
	Error	Distribution		
SCALE:C(10)	8.258432	0.554534	14.89256	0.0000

R-squared	0.151569	Mean dependent var	1.455907
Adjusted R-squared	0.138649	S.D. dependent var	3.298758
S.E. of regression	3.061544	Akaike info criterion	2.378473
Sum squared resid	5539.472	Schwarz criterion	2.451661
Log likelihood	-704.7311	Hannan-Quinn criter.	2.406961
Avg. log likelihood	-1.172597		
Left censored obs	451	Right censored obs	0
Uncensored obs	150	Total obs	601

Tests of Significance

EViews does not, by default, provide you with the usual likelihood ratio test of the overall significance for the tobit and other censored regression models. There are several ways to perform this test (or an asymptotically equivalent test).

First, you can use the built-in coefficient testing procedures to test the exclusion of all of the explanatory variables. Select the redundant variables test and enter the names of all of the explanatory variables you wish to exclude. EViews will compute the appropriate likelihood ratio test statistic and the *p*-value associated with the statistic.

To take an example, suppose we wish to test whether the variables in the Fair tobit, above, contribute to the fit of the model. Select **View/Coefficient Tests/Redundant Variables - Likelihood Ratio...** and enter all of the explanatory variables:

```
z1 z2 z3 z4 z5 z6 z7 z8
```

EViews will estimate the restricted model for you and compute the LR statistic and *p*-value. In this case, the value of the test statistic is 80.01, which for eight degrees of freedom, yields a *p*-value of less than 0.000001.

Alternatively, you could test the restriction using the Wald test by selecting **View/Coefficient Tests/Wald - Coefficient Restrictions...**, and entering the restriction that:

```
c(2)=c(3)=c(4)=c(5)=c(6)=c(7)=c(8)=c(9)=0
```

The reported statistic is 68.14, with a *p*-value of less than 0.000001.

Lastly, we demonstrate the direct computation of the LR test. Suppose the Fair tobit model estimated above is saved in the named equation EQ_TOBIT. Then you could estimate an equation containing only a constant, say EQ_RESTR, and place the likelihood ratio statistic in a scalar:

```
scalar lrstat=-2*(eq_restr.@logl-eq_tobit.@logl)
```

Next, evaluate the chi-square probability associated with this statistic:

```
scalar lrprob=1-@cchisq(lrstat, 8)
```

with degrees of freedom given by the number of coefficient restrictions in the constant only model. You can double click on the LRSTAT icon or the LRPROB icon in the workfile window to display the results in the status line.

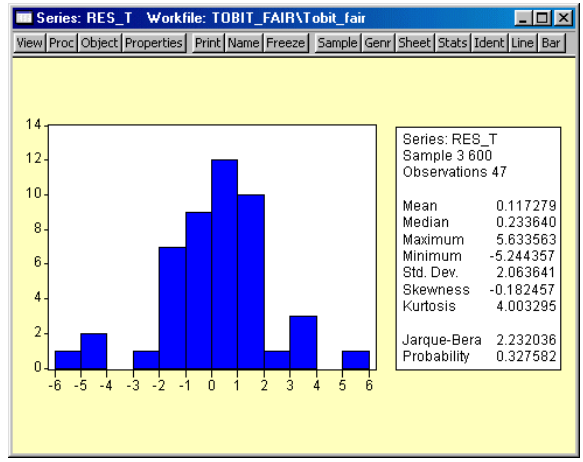
A Specification Test for the Tobit

As a rough diagnostic check, Pagan and Vella (1989) suggest plotting Powell's (1986) symmetrically trimmed residuals. If the error terms have a symmetric distribution centered at zero (as assumed by the normal distribution), so should the trimmed residuals. To con-

struct the trimmed residuals, first save the forecasts of the index (expected latent variable): click **Forecast**, choose **Index-Expected latent variable**, and provide a name for the fitted index, say XB. The trimmed residuals are obtained by dropping observations for which $x_i'\hat{\beta} < 0$, and replacing y_i with $2(x_i'\hat{\beta})$ for all observations where $y_i < 2(x_i'\hat{\beta})$. The trimmed residuals RES_T can be obtained by using the commands:

```
series res_t=(y_pt<=2*xb)*(y_pt-xb) +(y_pt>2*xb)*xb
smpl if xb<0
series res_t=na
smpl @all
```

The histogram of the trimmed residual is depicted below.



This example illustrates the possibility that the number of observations that are lost by trimming can be quite large; out of the 601 observations in the sample, only 47 observations are left after trimming.

The tobit model imposes the restriction that the coefficients that determine the probability of being censored are the same as those that determine the conditional mean of the uncensored observations. To test this restriction, we carry out the LR test by comparing the (restricted) tobit to the unrestricted log likelihood that is the sum of a probit and a truncated regression (we discuss truncated regression in detail in the following section). Save the tobit equation in the workfile by pressing the **Name** button, and enter a name, say EQ_TOBIT.

To estimate the probit, first create a dummy variable indicating uncensored observations by the command:

```
series y_c = (y_pt>0)
```

Then estimate a probit by replacing the dependent variable Y_PT by Y_C. A simple way to do this is to press **Object/Copy Object…** from the tobit equation toolbar. From the new untitled equation window that appears, press **Estimate**, replace the dependent variable with Y_C and choose **Method: BINARY** and click **OK**. Save the probit equation by pressing the **Name** button, say as EQ_BIN.

To estimate the truncated model, press **Object/Copy Object...** from the tobit equation toolbar again. From the new untitled equation window that appears, press **Estimate**, mark the **Truncated sample** option, and click **OK**. Save the truncated regression by pressing the **Name** button, say as EQ_TR.

Then the LR test statistic and its p-value can be saved as a scalar by the commands:

```
scalar lr_test=2*(eq_bin.@logl+eq_tr.@logl-eq_tobit.@logl)
scalar lr_pval=1-@cchisq(lr_test,eq_tobit.@ncoef)
```

Double click on the scalar name to display the value in the status line at the bottom of the EViews window. For the example data set, the p-value is 0.066, which rejects the tobit model at the 10% level, but not at the 5% level.

For other specification tests for the tobit, see Greene (1997, 20.3.4) or Pagan and Vella (1989).

Truncated Regression Models

A close relative of the censored regression model is the truncated regression model. Suppose that an observation is not observed whenever the dependent variable falls below one threshold, or exceeds a second threshold. This sampling rule occurs, for example, in earnings function studies for low-income families that exclude observations with incomes above a threshold, and in studies of durables demand among individuals who purchase durables.

The general two-limit truncated regression model may be written as:

$$y_i^* = x_i'\beta + \epsilon_i \tag{21.32}$$

where $y_i = y_i^*$ is only observed if:

$$\underline{c}_i < y_i^* < \bar{c}_i . \tag{21.33}$$

If there is no lower truncation, then we can set $\underline{c}_i = -\infty$. If there is no upper truncation, then we set $\bar{c}_i = \infty$.

The log likelihood function associated with these data is given by:

$$l(\beta, \sigma) = \sum_{i=1}^{N} \log f((y_i - x_i'\beta)/\sigma) \cdot 1(\underline{c}_i < y_i < \bar{c}_i) \tag{21.34}$$
$$- \log (F((\bar{c}_i - x_i'\beta)/\sigma) - F((\underline{c}_i - x_i'\beta)/\sigma)).$$

The likelihood function is maximized with respect to β and σ, using standard iterative methods.

Estimating a Truncated Model in EViews

Estimation of a truncated regression model follows the same steps as estimating a censored regression:

- Select **Quick/Estimate Equation…** from the main menu, and in the Equation Specification dialog, select the **CENSORED** estimation method. The censored and truncated regression dialog will appear.
- Enter the name of the truncated dependent variable and the list of the regressors in the **Equation Specification** field, and select one of the three distributions for the error term. You must enter your specification by list.
- Indicate that you wish to estimate the truncated model by checking the **Truncated sample** option.
- Specify the truncation points of the dependent variable by entering the appropriate expressions in the two edit fields. If you leave an edit field blank, EViews will assume that there is no truncation along that dimension.

You should keep a few points in mind. First, truncated estimation is only available for models where the truncation points are known, since the likelihood function is not otherwise defined. If you attempt to specify your truncation points by index, EViews will issue an error message indicating that this selection is not available.

Second, EViews will issue an error message if any values of the dependent variable are outside the truncation points. Furthermore, EViews will automatically exclude any observations that are exactly equal to a truncation point. Thus, if you specify zero as the lower truncation limit, EViews will issue an error message if any observations are less than zero, and will exclude any observations where the dependent variable exactly equals zero.

The cumulative distribution function and density of the assumed distribution will be used to form the likelihood function, as described above.

Procedures for Truncated Equations

EViews provides the same procedures for truncated equations as for censored equations. The residual and forecast calculations differ to reflect the truncated dependent variable and the different likelihood function.

Make Residual Series

Select **Proc/Make Residual Series**, and select from among the three types of residuals. The three types of residuals for censored models are defined as:

Ordinary	$e_{oi} = y_i - \mathrm{E}(y_i^* \mid \underline{c}_i < y_i^* < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma})$
Standardized	$e_{si} = \dfrac{y_i - \mathrm{E}(y_i^* \mid \underline{c}_i < y_i^* < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma})}{\sqrt{\mathrm{var}(y_i^* \mid \underline{c}_i < y_i^* < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma})}}$
Generalized	$e_{gi} = -\dfrac{f'((y_i - x_i'\hat{\beta})/\hat{\sigma})}{\sigma f((y_i - x_i'\hat{\beta})/\hat{\sigma})} - \dfrac{f((\bar{c}_i - x_i'\hat{\beta})/\hat{\sigma}) - f((\underline{c}_i - x_i'\hat{\beta})/\hat{\sigma})}{\sigma(F((\bar{c}_i - x_i'\hat{\beta})/\hat{\sigma}) - F((\underline{c}_i - x_i'\hat{\beta})/\hat{\sigma}))}$

where f, F, are the density and distribution functions. Details on the computation of $\mathrm{E}(y_i \mid \underline{c}_i < y_i < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma})$ are provided below.

The generalized residuals may be used as the basis of a number of LM tests, including LM tests of normality (see Chesher and Irish (1984, 1987), and Gourieroux, Monfort and Trognon (1987); Greene (1997) provides a brief discussion and additional references).

Forecasting

EViews provides you with the option of forecasting the expected observed dependent variable, $\mathrm{E}(y_i \mid x_i, \hat{\beta}, \hat{\sigma})$, or the expected latent variable, $\mathrm{E}(y_i^* \mid x_i, \hat{\beta}, \hat{\sigma})$.

To forecast the expected latent variable, select **Forecast** from the equation toolbar to open the forecast dialog, click on **Index - Expected latent variable,** and enter a name for the series to hold the output. The forecasts of the expected latent variable $\mathrm{E}(y_i^* \mid x_i, \hat{\beta}, \hat{\sigma})$ are computed using:

$$\hat{y}_i^* = \mathrm{E}(y_i^* \mid x_i, \hat{\beta}, \hat{\sigma}) = x_i'\hat{\beta}. \tag{21.35}$$

To forecast the expected observed dependent variable for the truncated model, you should select **Expected dependent variable**, and enter a series name. These forecasts are computed using:

$$\hat{y}_i = \mathrm{E}(y_i^* \mid \underline{c}_i < y_i^* < \bar{c}_i; x_i, \hat{\beta}, \hat{\sigma}) \tag{21.36}$$

so that the expectations for the latent variable are taken with respect to the conditional (on being observed) distribution of the y_i^*. Note that these forecasts always satisfy the inequality $\underline{c}_i < \hat{y}_i < \bar{c}_i$.

It is instructive to compare this latter expected value with the expected value derived for the censored model in Equation (21.30) above (repeated here for convenience):

$$\begin{aligned}\hat{y}_i &= \mathrm{E}(y_i | x_i, \hat{\beta}, \hat{\sigma}) = \underline{c}_i \cdot \Pr(y_i = \underline{c}_i | x_i, \hat{\beta}, \hat{\sigma}) \\ &+ \mathrm{E}(y_i^* | \underline{c}_i < y_i^* < \bar{c}_i;\ x_i, \hat{\beta}, \hat{\sigma}) \cdot \Pr(\underline{c}_i < y_i^* < \bar{c}_i | x_i, \hat{\beta}, \hat{\sigma}) \\ &+ \bar{c}_i \cdot \Pr(y_i = \underline{c}_i | x_i, \hat{\beta}, \hat{\sigma}).\end{aligned} \tag{21.37}$$

The expected value of the dependent variable for the truncated model is the first part of the middle term of the censored expected value. The differences between the two expected values (the probability weight and the first and third terms) reflect the different treatment of latent observations that do not lie between $\underline{c}_i$ and $\bar{c}_i$. In the censored case, those observations are included in the sample and are accounted for in the expected value. In the truncated case, data outside the interval are not observed and are not used in the expected value computation.

Illustration

As an example, we reestimate the Fair tobit model from above, truncating the data so that observations at or below zero are removed from the sample. The output from truncated estimation of the Fair model is presented below:

Dependent Variable: Y_PT
Method: ML - Censored Normal (TOBIT)
Date: 10/13/97 Time: 22:45
Sample(adjusted): 452 601
Included observations: 150 after adjusting endpoints
Truncated sample
Left censoring (value) at zero
Convergence achieved after 8 iterations
Covariance matrix computed using second derivatives

	Coefficient	Std. Error	z-Statistic	Prob.
C	12.37288	5.178306	2.389368	0.0169
Z1	-1.336872	1.453133	-0.919993	0.3576
Z2	-0.044792	0.116141	-0.385670	0.6997
Z3	0.544182	0.220119	2.472218	0.0134
Z4	-2.142896	1.787720	-1.198675	0.2307
Z5	-1.423128	0.600472	-2.370014	0.0178
Z6	-0.316721	0.322327	-0.982609	0.3258
Z7	0.621428	0.478827	1.297813	0.1944
Z8	-1.210037	0.552131	-2.191578	0.0284
Error Distribution				
SCALE:C(10)	5.379557	0.688875	7.809196	0.0000

R-squared	0.654664	Mean dependent var	1.455907
Adjusted R-squared	0.649405	S.D. dependent var	3.298758
S.E. of regression	1.953229	Akaike info criterion	1.333891
Sum squared resid	2254.726	Schwarz criterion	1.407079
Log likelihood	-390.8342	Hannan-Quinn criter.	1.362379
Avg. log likelihood	-0.650306		

Left censored obs	0	Right censored obs	0
Uncensored obs	150	Total obs	150

Note that the header information indicates that the model is a truncated specification, and that the sample information at the bottom of the screen shows that there are no left and right censored observations.

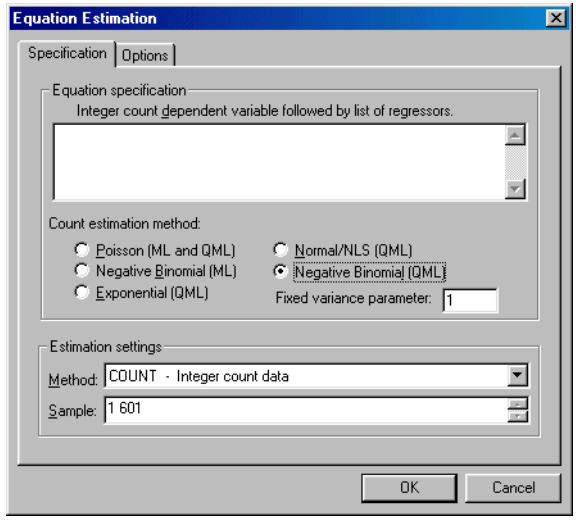
Count Models

Count models are employed when y takes integer values that represent the number of events that occur—examples of count data include the number of patents filed by a company, and the number of spells of unemployment experienced over a fixed time interval.

EViews provides support for the estimation of several models of count data. In addition to the standard poisson and negative binomial maximum likelihood (ML) specifications, EViews provides a number of quasi-maximum likelihood (QML) estimators for count data.

Estimating Count Models in EViews

To estimate a count data model, select **Quick/Estimate Equation…** from the main menu, and select **COUNT** as the estimation method. EViews displays the count estimation dialog into which you will enter the dependent and explanatory variable regressors, select a type of count model, and if desired, set estimation options.



There are three parts to the specification of the count model:

- In the upper edit field, you should list the dependent variable and the independent variables. You must specify your model by list. The list of explanatory variables specifies a model for the conditional mean of the dependent variable:

$$m(x_i, \beta) = \mathrm{E}(y_i | x_i, \beta) = \exp(x_i'\beta). \tag{21.38}$$

- Next, click on **Options** and, if desired, change the default estimation algorithm, convergence criterion, starting values, and method of computing the coefficient covariance.
- Lastly, select one of the entries listed under count estimation method, and if appropriate, specify a value for the variance parameter. Details for each method are provided in the following discussion.

Poisson Model

For the Poisson model, the conditional density of y_i given x_i is:

$$f(y_i|x_i, \beta) = e^{-m(x_i, \beta)} m(x_i, \beta)^{y_i} / y_i! \tag{21.39}$$

where y_i is a non-negative integer valued random variable. The maximum likelihood estimator (MLE) of the parameter β is obtained by maximizing the log likelihood function:

$$l(\beta) = \sum_{i=1}^{N} y_i \log m(x_i, \beta) - m(x_i, \beta) - \log(y_i!) . \tag{21.40}$$

Provided the conditional mean function is correctly specified and the conditional distribution of y is Poisson, the MLE $\hat{\beta}$ is consistent, efficient, and asymptotically normally distributed, with variance matrix consistently estimated by:

$$V = \text{var}(\hat{\beta}) = \left(\sum_{i=1}^{N} \left(\frac{\partial \hat{m}_i}{\partial \beta} \frac{\partial \hat{m}_i}{\partial \beta'} \right) / \hat{m}_i \right)^{-1} \tag{21.41}$$

where $\hat{m}_i = m(x_i, \hat{\beta})$.

The Poisson assumption imposes restrictions that are often violated in empirical applications. The most important restriction is the equality of the (conditional) mean and variance:

$$v(x_i, \beta) = \text{var}(y_i|x_i, \beta) = \text{E}(y_i|x_i, \beta) = m(x_i, \beta) . \tag{21.42}$$

If the mean-variance equality does not hold, the model is misspecified. EViews provides a number of other estimators for count data which relax this restriction.

We note here that the Poisson estimator may also be interpreted as a quasi-maximum likelihood estimator. The implications of this result are discussed below.

Negative Binomial (ML)

One common alternative to the Poisson model is to estimate the parameters of the model using maximum likelihood of a negative binomial specification. The log likelihood for the negative binomial distribution is given by:

$$
\begin{aligned}
l(\beta, \eta) \; = \; & \sum_{i=1}^{N} y_i \log(\eta^2 m(x_i, \beta)) \\
& - (y_i + 1/\eta^2)\log(1 + \eta^2 m(x_i, \beta)) \\
& + \log\Gamma(y_i + 1/\eta^2) - \log(y_i!) - \log\Gamma(1/\eta^2)
\end{aligned}
\tag{21.43}
$$

where η^2 is a variance parameter to be jointly estimated with the conditional mean parameters β. EViews estimates the log of η^2, and labels this parameter as the "SHAPE" parameter in the output. Standard errors are computed using the inverse of the information matrix.

The negative binomial distribution is often used when there is *overdispersion* in the data, so that $v(x_i, \beta) > m(x_i, \beta)$, since the following moment conditions hold:

$$
\begin{aligned}
\mathrm{E}(y_i | x_i, \beta) \; &= \; m(x_i, \beta) \\
\mathrm{var}(y_i | x_i, \beta) \; &= \; m(x_i, \beta)(1 + \eta^2 m(x_i, \beta))
\end{aligned}
\tag{21.44}
$$

η^2 is therefore a measure of the extent to which the conditional variance exceeds the conditional mean.

Consistency and efficiency of the negative binomial ML requires that the conditional distribution of y be negative binomial.

Quasi-maximum Likelihood (QML)

We can perform maximum likelihood estimation under a number of alternative distributional assumptions. These *quasi-maximum likelihood (QML) estimators* are robust in the sense that they produce consistent estimates of the parameters of a correctly specified conditional mean, even if the distribution is incorrectly specified.

This robustness result is exactly analogous to the situation in ordinary regression, where the normal ML estimator (least squares) is consistent, even if the underlying error distribution is not normally distributed. In ordinary least squares, all that is required for consistency is a correct specification of the conditional mean $m(x_i, \beta) \; = \; x_i'\beta$. For QML count models, all that is required for consistency is a correct specification of the conditional mean $m(x_i, \beta)$.

The estimated standard errors computed using the inverse of the information matrix will not be consistent unless the conditional distribution of y is correctly specified. However, it is possible to estimate the standard errors in a robust fashion so that we can conduct valid inference, even if the distribution is incorrectly specified.

EViews provides options to compute two types of robust standard errors. Click **Options** in the Equation Specification dialog box and mark the **Robust Covariance** option. The **Huber/White** option computes QML standard errors, while the **GLM** option computes standard errors corrected for overdispersion. See "Technical Notes" on page 667 for details on these options.

Further details on QML estimation are provided by Gourioux, Monfort, and Trognon (1994a, 1994b). Wooldridge (1996) provides an excellent summary of the use of QML techniques in estimating parameters of count models. See also the extensive related literature on Generalized Linear Models (McCullagh and Nelder, 1989).

Poisson

The Poisson MLE is also a QMLE for data from alternative distributions. Provided that the conditional mean is correctly specified, it will yield consistent estimates of the parameters β of the mean function. By default, EViews reports the ML standard errors. If you wish to compute the QML standard errors, you should click on **Options**, select **Robust Covariances**, and select the desired covariance matrix estimator.

Exponential :

The log likelihood for the exponential distribution is given by:

$$l(\beta) = \sum_{i=1}^{N} -\log m(x_i, \beta) - y_i/(m(x_i, \beta)) . \tag{21.45}$$

As with the other QML estimators, the exponential QMLE is consistent even if the conditional distribution of y_i is not exponential, provided that m_i is correctly specified. By default, EViews reports the robust QML standard errors.

Normal

The log likelihood for the normal distribution is:

$$l(\beta) = \sum_{i=1}^{N} -\frac{1}{2}\left(\frac{y_i - m(x_i, \beta)}{\sigma}\right)^2 - \frac{1}{2}\log(\sigma^2) - \frac{1}{2}\log(2\pi) . \tag{21.46}$$

For *fixed* σ^2 and correctly specified m_i, maximizing the normal log likelihood function provides consistent estimates even if the distribution is not normal. Note that maximizing the normal log likelihood for a fixed σ^2 is equivalent to minimizing the sum of squares for the nonlinear regression model:

$$y_i = m(x_i, \beta) + \epsilon_i . \tag{21.47}$$

EViews sets $\sigma^2 = 1$ by default. You may specify any other (positive) value for σ^2 by changing the number in the **Fixed variance parameter** field box. By default, EViews reports the robust QML standard errors when estimating this specification.

Negative Binomial

If we maximize the negative binomial log likelihood, given above, for *fixed* η^2, we obtain the QMLE of the conditional mean parameters β. This QML estimator is consistent even if the conditional distribution of y is not negative binomial, provided that m_i is correctly specified.

EViews sets $\eta^2 = 1$ by default, which is a special case known as the geometric distribution. You may specify any other (positive) value by changing the number in the **Fixed variance parameter** field box. For the negative binomial QMLE, EViews by default reports the robust QMLE standard errors.

Views of Count Models

EViews provides a full complement of views of count models. You can examine the estimation output, compute frequencies for the dependent variable, view the covariance matrix, or perform coefficient tests. Additionally, you can select **View/Actual, Fitted, Residual...** and pick from a number of views describing the ordinary residuals $e_{oi} = y_i - m(x_i, \hat{\beta})$, or you can examine the correlogram and histogram of these residuals. For the most part, all of these views are self-explanatory.

Note, however, that the LR test statistics presented in the summary statistics at the bottom of the equation output, or as computed under the **View/Coefficient Tests/Redundant Variables - Likelihood Ratio...** have a known asymptotic distribution only if the conditional distribution is correctly specified. Under the weaker GLM assumption that the true variance is proportional to the nominal variance, we can form a quasi-likelihood ratio, $QLR = LR/\hat{\sigma}^2$, where $\hat{\sigma}^2$ is the estimated proportional variance factor. This QLR statistic has an asymptotic χ^2 distribution under the assumption that the mean is correctly specified and that the variances follow the GLM structure. EViews does not compute the QLR statistic, but it can be estimated by computing an estimate of $\hat{\sigma}^2$ based upon the standardized residuals. We provide an example of the use of the QLR test statistic below.

If the GLM assumption does not hold, then there is no usable QLR test statistic with a known distribution; see Wooldridge (1996).

Procedures for Count Models

Most of the procedures are self-explanatory. Some details are required for the forecasting and residual creation procedures.

- **Forecast...** provides you the option to forecast the dependent variable y_i or the predicted linear index $x_i'\hat{\beta}$. Note that for all of these models the forecasts of y_i are given by $\hat{y}_i = m(x_i, \hat{\beta})$ where $m(x_i, \hat{\beta}) = \exp(x_i'\hat{\beta})$.

- **Make Residual Series…** provides the following three types of residuals for count models:

Ordinary	$e_{oi} = y_i - m(x_i, \hat{\beta})$
Standardized (Pearson)	$e_{si} = \frac{y_i - m(x_i, \hat{\beta})}{\sqrt{v(x_i, \hat{\beta}, \hat{\gamma})}}$
Generalized	e_g = (varies)

where the γ represents any additional parameters in the variance specification. Note that the specification of the variances may vary significantly between specifications. For example, the Poisson model has $v(x_i, \hat{\beta}) = m(x_i, \hat{\beta})$, while the exponential has $v(x_i, \hat{\beta}) = m(x_i, \hat{\beta})^2$.

The generalized residuals can be used to obtain the score vector by multiplying the generalized residuals by each variable in x. These scores can be used in a variety of LM or conditional moment tests for specification testing; see Wooldridge (1996).

Demonstrations

A Specification Test for Overdispersion

Consider the model:

$$\text{NUMB}_i = \beta_1 + \beta_2 \text{IP}_i + \beta_3 \text{FEB}_i + \epsilon_i, \tag{21.48}$$

where the dependent variable NUMB is the number of strikes, IP is a measure of industrial production, and FEB is a February dummy variable, as reported in Kennan (1985, Table 1).

The results from Poisson estimation of this model are presented below:

Dependent Variable: NUMB
Method: ML/QML - Poisson Count
Date: 09/14/97 Time: 10:58
Sample: 1 103
Included observations: 103
Convergence achieved after 4 iterations
Covariance matrix computed using second derivatives

Variable	Coefficient	Std. Error	z-Statistic	Prob.
C	1.725630	0.043656	39.52764	0.0000
IP	2.775334	0.819104	3.388254	0.0007
FEB	-0.377407	0.174520	-2.162539	0.0306

R-squared	0.064502	Mean dependent var	5.495146
Adjusted R-squared	0.045792	S.D. dependent var	3.653829
S.E. of regression	3.569190	Akaike info criterion	5.583421
Sum squared resid	1273.912	Schwarz criterion	5.660160
Log likelihood	-284.5462	Hannan-Quinn criter.	5.614503
Restr. log likelihood	-292.9694	Avg. log likelihood	-2.762584
LR statistic (2 df)	16.84645	LR index (Pseudo-R2)	0.028751
Probability(LR stat)	0.000220		

Cameron and Trivedi (1990) propose a regression based test of the Poisson restriction $v(x_i, \beta) = m(x_i, \beta)$. To carry out the test, first estimate the Poisson model and obtain the fitted values of the dependent variable. Click **Forecast** and provide a name for the forecasted dependent variable, say NUMB_F. The test is based on an auxiliary regression of $e_{oi}^2 - y_i$ on $\hat{y}_i^2$ and testing the significance of the regression coefficient. For this example, the test regression can be estimated by the command:

```
ls (numb-numb_f)^2-numb numb_f^2
```

yielding the following results:

Dependent Variable: (NUMB-NUMB_F)^2-NUMB
Method: Least Squares
Date: 09/14/97 Time: 11:05
Sample: 1 103
Included observations: 103

Variable	Coefficient	Std. Error	t-Statistic	Prob.
NUMB_F^2	0.238874	0.052115	4.583571	0.0000

R-squared	0.043930	Mean dependent var	6.872929
Adjusted R-squared	0.043930	S.D. dependent var	17.65726
S.E. of regression	17.26506	Akaike info criterion	8.544908
Sum squared resid	30404.41	Schwarz criterion	8.570488
Log likelihood	-439.0628	Durbin-Watson stat	1.711805

The t-statistic of the coefficient is highly significant, leading us to reject the Poisson restriction. Moreover, the estimated coefficient is significantly positive, indicating overdispersion in the residuals.

An alternative approach, suggested by Wooldridge (1996), is to regress $e_{si} - 1$, on $\hat{y}_i$. To perform this test, select **Proc/Make Residual Series...** and select **Standardized**. Save the results in a series, say SRESID. Then estimating the regression specification:

```
sresid^2-1 numbf
```

yields the results:

Dependent Variable: SRESID^2-1
Method: Least Squares
Date: 10/06/97 Time: 16:05
Sample: 1 103
Included observations: 103

Variable	Coefficient	Std. Error	t-Statistic	Prob.
NUMBF	0.221238	0.055002	4.022326	0.0001

R-squared	0.017556	Mean dependent var	1.161573
Adjusted R-squared	0.017556	S.D. dependent var	3.138974
S.E. of regression	3.111299	Akaike info criterion	5.117619
Sum squared resid	987.3786	Schwarz criterion	5.143199
Log likelihood	-262.5574	Durbin-Watson stat	1.764537

Both tests suggest the presence of overdispersion, with the variance approximated by $v = m(1 + 0.23m)$.

Given the evidence of overdispersion and the rejection of the Poisson restriction, we will re-estimate the model, allowing for mean-variance inequality. Our approach will be to estimate the two-step negative binomial QMLE specification (termed the *quasi-generalized pseudo-maximum likelihood estimator* by Gourieroux, Monfort, and Trognon (1984a, b)) using the estimate of $\hat{\eta}^2$ derived above. To compute this estimator, simply select **Negative Binomial (QML)** and enter 0.22124 in the edit field for **Fixed variance parameter**.

We will use the GLM variance calculations, so you should click on **Option** in the Equation Specification dialog and mark the **Robust Covariance** and **GLM** options. The estimation results are shown below:

Dependent Variable: NUMB
Method: QML - Negative Binomial Count
Date: 10/11/97 Time: 23:53
Sample: 1 103
Included observations: 103
QML parameter used in estimation: 0.22124
Convergence achieved after 3 iterations
GLM Robust Standard Errors & Covariance
Variance factor estimate = 2.465660162
Covariance matrix computed using second derivatives

Variable	Coefficient	Std. Error	z-Statistic	Prob.
C	1.724906	0.102543	16.82135	0.0000
IP	2.833103	1.919447	1.475999	0.1399
FEB	-0.369558	0.377376	-0.979285	0.3274

R-squared	0.064374	Mean dependent var	5.495146
Adjusted R-squared	0.045661	S.D. dependent var	3.653829
S.E. of regression	3.569435	Akaike info criterion	5.174385
Sum squared resid	1274.087	Schwarz criterion	5.251125
Log likelihood	-263.4808	Hannan-Quinn criter.	5.205468
Restr. log likelihood	-522.9973	Avg. log likelihood	-2.558066
LR statistic (2 df)	519.0330	LR index (Pseudo-R2)	0.496210
Probability(LR stat)	0.000000		

The header indicates that the estimated GLM variance factor is 2.4, suggesting that the negative binomial ML would not have been an appropriate specification. Nevertheless, the negative binomial QML should be consistent, and under the GLM assumption, the standard errors should be consistently estimated. It is worth noting that the coefficients on IP and FEB, which were strongly statistically significant in the Poisson specification, are no longer significantly different from zero at conventional significance levels.

Quasi-likelihood Ratio Statistic

As described by Wooldridge (1996), specification testing using likelihood ratio statistics requires some care when based upon QML models. We illustrate here the differences between a standard LR test for significant coefficients and the corresponding QLR statistic.

From the results above, we know that the overall likelihood ratio statistic for the Poisson model is 16.85, with a corresponding p-value of 0.0002. This statistic is valid under the assumption that $m(x_i, \beta)$ is specified correctly and that the mean-variance equality holds.

We can decisively reject the latter hypothesis, suggesting that we should derive the QML estimator with consistently estimated covariance matrix under the GLM variance assumption. While EViews does not automatically adjust the LR statistic to reflect the QML assumption, it is easy enough to compute the adjustment by hand. Following Wooldridge, we construct the QLR statistic by dividing the original LR statistic by the estimated GLM variance factor.

Suppose that the estimated QML equation is named EQ1. Then you can use EViews to compute p-value associated with this statistic, placing the results in scalars using the following commands:

```
scalar qlr = eq1.@logl/2.226420477
scalar qpval = 1-@cchisq(qlr, 2)
```

You can examine the results by clicking on the scalar objects in the workfile window and viewing the results in the status line. The QLR statistic is 7.5666, and the p-value is 0.023. The statistic and p-value are valid under the weaker conditions that the conditional mean is correctly specified, and that the conditional variance is proportional (but not necessarily equal) to the conditional mean.

Technical Notes

Huber/White (QML) Standard Errors

The Huber/White options for robust standard errors computes the quasi-maximum likelihood (or pseudo-ML) standard errors:

$$\text{var}_{QML}(\hat{\beta}) = \hat{H}^{-1}\hat{g}\hat{g}'\hat{H}^{-1}, \tag{21.49}$$

where $\hat{g}$ and $\hat{H}^{-1}$ are the gradient (or score) and Hessian of the log likelihood evaluated at the ML estimates.

Note that these standard errors are *not* robust to heteroskedasticity in binary dependent variable models. They are robust to certain misspecifications of the underlying distribution of y.

GLM Standard Errors

Many of the discrete and limited dependent variable models described in this chapter belong to a class of models known as *generalized linear models* (GLM). The assumption of GLM is that the distribution of the dependent variable y_i belongs to the exponential family and that the conditional mean of y_i is a (smooth) nonlinear transformation of the linear part $x_i'\beta$:

$$\text{E}(y_i|x_i, \beta) = h(x_i'\beta). \tag{21.50}$$

Even though the QML covariance is robust to general misspecification of the conditional distribution of y_i, it does not possess any efficiency properties. An alternative consistent estimate of the covariance is obtained if we impose the GLM condition that the (true) variance of y_i is proportional to the variance of the distribution used to specify the log likelihood:

$$\text{var}(y_i|x_i, \beta) = \sigma^2\text{var}_{ML}(y_i|x_i, \beta). \tag{21.51}$$

In other words, the ratio of the (conditional) variance to the mean is some constant σ^2 that is independent of x. The most empirically relevant case is $\sigma^2 > 1$, which is known as *overdispersion*. If this proportional variance condition holds, a consistent estimate of the GLM covariance is given by:

$$\text{var}_{GLM}(\hat{\beta}) = \hat{\sigma}^2 \text{var}_{ML}(\hat{\beta}), \tag{21.52}$$

where

$$\hat{\sigma}^2 = \frac{1}{N-K} \cdot \sum_{i=1}^{N} \frac{(y_i - \hat{y}_i)^2}{\sqrt{v(x_i, \hat{\beta}, \hat{\gamma})}} = \frac{1}{N-K} \cdot \sum_{i=1}^{N} \frac{\hat{u}_i^2}{\sqrt{(v(x_i, \hat{\beta}, \hat{\gamma}))}}. \tag{21.53}$$

If you select GLM standard errors, the estimated proportionality term $\hat{\sigma}^2$ is reported as the variance factor estimate in EViews.

For more discussion on GLM and the phenomenon of overdispersion, see McCullaugh and Nelder (1989) or Fahrmeir and Tutz (1994).

The Hosmer-Lemeshow Test

Let the data be grouped into $j = 1, 2, \ldots, J$ groups, and let n_j be the number of observations in group j. Define the number of $y_i = 1$ observations and the average of predicted values in group j as:

$$\begin{aligned} y(j) &= \sum_{i \in j} y_i \\ \bar{p}(j) &= \sum_{i \in j} \hat{p}_i / n_j = \sum_{i \in j} (1 - F(-x_i'\hat{\beta}))/n_j \end{aligned} \tag{21.54}$$

The Hosmer-Lemeshow test statistic is computed as:

$$HL = \sum_{j=1}^{J} \frac{(y(j) - n_j \bar{p}(j))^2}{n_j \bar{p}(j)(1 - \bar{p}(j))}. \tag{21.55}$$

The distribution of the HL statistic is not known; however, Hosmer and Lemeshow (1989, p.141) report evidence from extensive simulation indicating that when the model is correctly specified, the distribution of the statistic is well approximated by a χ^2 distribution with $J-2$ degrees of freedom. Note that these findings are based on a simulation where J is close to n.

The Andrews Test

Let the data be grouped into $j = 1, 2, \ldots, J$ groups. Since y is binary, there are $2J$ cells into which any observation can fall. Andrews (1988a, 1988b) compares the $2J$ vector of the actual number of observations in each cell to those predicted from the model, forms a quadratic form, and shows that the quadratic form has an asymptotic χ^2 distribution if the model is specified correctly.

Andrews suggests three tests depending on the choice of the weighting matrix in the quadratic form. EViews uses the test that can be computed by an auxiliary regression as described in Andrews (1988a, 3.18) or Andrews (1988b, 17).

Briefly, let $\tilde{A}$ be an $n \times J$ matrix with element $\tilde{a}_{ij} = 1(i \in j) - \hat{p}_i$, where the indicator function $1(i \in j)$ takes the value one if observation i belongs to group j with $y_i = 1$, and zero otherwise (we drop the columns for the groups with $y = 0$ to avoid singularity). Let B be the $n \times K$ matrix of the contributions to the score $\partial l(\beta)/\partial \beta'$. The Andrews test statistic is n times the R^2 from regressing a constant (one) on each column of $\tilde{A}$ and B. Under the null hypothesis that the model is correctly specified, nR^2 is asymptotically distributed χ^2 with J degrees of freedom.

Chapter 22. The Log Likelihood (LogL) Object

EViews contains customized procedures which help solve the majority of the estimation problems that you might encounter. On occasion, however, you may come across an estimation specification which is not included among these specialized routines. This specification may be an extension of an existing procedure, or it could be an entirely new class of problem.

Fortunately, EViews provides you with tools to estimate a wide variety of specifications through the *log likelihood (logl)* object. The logl object provides you with a general, open-ended tool for estimating a broad class of specifications by maximizing a likelihood function with respect to parameters.

When working with a log likelihood object, you will use EViews' series generation capabilities to describe the log likelihood contribution of each observation in your sample as a function of unknown parameters. You may supply analytical derivatives of the likelihood for one or more parameters, or you can simply let EViews calculate numeric derivatives automatically. EViews will search for the parameter values that maximize the specified likelihood function, and will provide estimated standard errors for these parameter estimates.

In this chapter, we provide an overview and describe the general features of the logl object. We also give examples of specifications which may be estimated using the object. The examples include: multinomial logit, unconditional maximum likelihood AR(1) estimation, Box-Cox regression, disequilibrium switching models, least squares with multiplicative heteroskedasticity, probit specifications with heteroskedasticity, probit with grouped data, nested logit, zero-altered Poisson models, Heckman sample selection models, Weibull hazard models, GARCH(1,1) with t-distributed errors, GARCH with coefficient restrictions, EGARCH with a generalized error distribution, and multivariate GARCH.

Overview

Most of the work in estimating a model using the logl object is in creating the text specification which will be used to evaluate the likelihood function.

If you are familiar with the process of generating series in EViews, you should find it easy to work with the logl specification, since the likelihood specification is merely a list of series assignment statements which are evaluated iteratively during the course of the maximization procedure. All you need to do is write down a set of statements which, when evaluated, will describe a series containing the contributions of each observation to the log likelihood function.

To take a simple example, suppose you believe that your data are generated by the conditional heteroskedasticity regression model:

$$\begin{aligned} y_t &= \beta_1 + \beta_2 x_t + \beta_3 z_t + \epsilon_t \\ \epsilon_t &\sim N(0, \sigma^2 z_t^{\alpha}) \end{aligned} \tag{22.1}$$

where x, y, and z are the observed series (data) and β_1, β_2, β_3, σ, α are the parameters of the model. The log likelihood function (the log of the density of the observed data) for a sample of T observations can be written as:

$$\begin{aligned} l(\beta, \alpha, \sigma) &= -\frac{T}{2}(\log(2\pi) + \log\sigma^2) - \frac{\alpha}{2}\sum_{t=1}^{T}\log(z_t) - \sum_{t=1}^{T}\frac{(y_t - \beta_1 - \beta_2 x_t - \beta_3 z_t)^2}{\sigma^2 z_t^{\alpha}} \\ &= \sum_{t=1}^{T}\left\{\log\phi\left(\frac{y_t - \beta_1 - \beta_2 x_t - \beta_3 z_t}{\sigma z_t^{\alpha/2}}\right) - \frac{1}{2}\log(\sigma^2 z_t^{\alpha})\right\} \end{aligned} \tag{22.2}$$

where ϕ is the standard normal density function.

Note that we can write the log likelihood function as a sum of the log likelihood contributions for each observation t:

$$l(\beta, \alpha, \sigma) = \sum_{t=1}^{T} l_t(\beta, \alpha, \sigma) \tag{22.3}$$

where the individual contributions are given by:

$$l_t(\beta, \alpha, \sigma) = \log\phi\left(\frac{y_t - \beta_1 - \beta_2 x_t - \beta_3 z_t}{\sigma z_t^{\alpha/2}}\right) - \frac{1}{2}\log(\sigma^2 z_t^{\alpha}) \tag{22.4}$$

Suppose that you know the true parameter values of the model, and you wish to generate a series in EViews which contains the contributions for each observation. To do this, you could assign the known values of the parameters to the elements C(1) to C(5) of the coefficient vector, and then execute the following list of assignment statements as commands or in an EViews program:

```
series res = y - c(1) - c(2)*x - c(3)*z
series var = c(4) * z^c(5)
series logl1 = log(@dnorm(res/@sqrt(var))) - log(var)/2
```

The first two statements describe series which will contain intermediate results used in the calculations. The first statement creates the residual series, RES, and the second statement creates the variance series, VAR. The series LOGL1 contains the set of log likelihood contributions for each observation.

Now suppose instead that you do not know the true parameter values of the model, and would like to estimate them from the data. The maximum likelihood estimates of the parameters are defined as the set of parameter values which produce the largest value of the likelihood function evaluated across all the observations in the sample.

The logl object makes finding these maximum likelihood estimates easy. Simply create a new log likelihood object, input the assignment statements above into the logl specification view, then ask EViews to estimate the specification.

In entering the assignment statements, you need only make two minor changes to the text above. First, the `series` keyword must be removed from the beginning of each line (since the likelihood specification implicitly assumes it is present). Second, an extra line must be added to the specification which identifies the name of the series in which the likelihood contributions will be contained. Thus, you should enter the following into your log likelihood object:

```
@logl logl1
res = y - c(1) - c(2)*x - c(3)*z
var = c(4) * z^c(5)
logl1 = log(@dnorm(res/@sqrt(var))) - log(var)/2
```

The first line in the log likelihood specification, `@logl logl1`, tells EViews that the series LOGL1 should be used to store the likelihood contributions. The remaining lines describe the computation of the intermediate results, and the actual likelihood contributions.

When you tell EViews to estimate the parameters of this model, it will execute the assignment statements in the specification repeatedly for different parameter values, using an iterative algorithm to search for the set of values that maximize the sum of the log likelihood contributions. When EViews can no longer improve the overall likelihood, it will stop iterating and will report final parameter values and estimated standard errors in the estimation output.

The remainder of this chapter discusses the rules for specification, estimation and testing using the likelihood object in greater detail.

Specification

To create a likelihood object, choose **Object/New Object…/LogL** or type "logl" in the command window. The likelihood window will open with a blank specification view. The specification view is a text window into which you enter a list of statements which describe your statistical model, and in which you set options which control various aspects of the estimation procedure.

Specifying the Likelihood

As described in the overview above, the core of the likelihood specification is a set of assignment statements which, when evaluated, generate a series containing the log likelihood contribution of each observation in the sample. There can be as many or as few of these assignment statements as you wish.

Each likelihood specification must contain a control statement which provides the name of the series which is used to contain the likelihood contributions. The format of this statement is:

```
@logl series_name
```

where `series_name` is the name of the series which will contain the contributions. This control statement may appear anywhere in the logl specification.

Whenever the specification is evaluated, whether for estimation or for carrying out a View or Proc, each assignment statement will be evaluated at the current parameter values, and the results stored in a series with the specified name. If the series does not exist, it will be created automatically. If the series already exists, EViews will use the existing series for storage, and will overwrite the data contained in the series.

If you would like to remove one or more of the series used in the specification after evaluation, you can use the `@temp` statement, as in:

```
@temp series_name1 series_name2
```

This statement tells EViews to delete any series in the list after evaluation of the specification is completed. Deleting these series may be useful if your logl creates a lot of intermediate results, and you do not want the series containing these results to clutter your workfile.

Parameter Names

In the example above, we used the coefficients C(1) to C(5) as names for our unknown parameters. More generally, any element of a named coefficient vector which appears in the specification will be treated as a parameter to be estimated.

In the conditional heteroskedasticity example, you might choose to use coefficients from three different coefficient vectors: one vector for the mean equation, one for the variance equation, and one for the variance parameters. You would first create three named coefficient vectors by the commands:

```
coef(3) beta
coef(1) scale
coef(1) alpha
```

You could then write the likelihood specification as:

```
@logl logl1
res = y - beta(1) - beta(2)*x - beta(3)*z
var = scale(1)*z^alpha(1)
logl1 = log(@dnorm(res/@sqrt(var))) - log(var)/2
```

Since all elements of named coefficient vectors in the specification will be treated as parameters, you should make certain that all coefficients really do affect the value of one or more of the likelihood contributions. If a parameter has no effect upon the likelihood, you will experience a singularity error when you attempt to estimate the parameters.

Note that all objects other than coefficient elements will be considered fixed and will not be updated during estimation. For example, suppose that SIGMA is a named scalar in your workfile. Then if you redefine the subexpression for VAR as:

```
var = sigma*z^alpha(1)
```

EViews will not estimate SIGMA. The value of SIGMA will remain fixed at its value at the start of estimation.

Order of Evaluation

The logl specification contains one or more assignment statements which generate the series containing the likelihood contributions. EViews always evaluates from top to bottom when executing these assignment statements, so expressions which are used in subsequent calculations should always be placed first.

EViews must also iterate through the observations in the sample. Since EViews iterates through both the equations in the specification and the observations in the sample, you will need to specify the order in which the evaluation of observations and equations occurs.

By default, EViews evaluates the specification *by observation* so that *all of the assignment statements* are evaluated for the *first observation*, then for the second observation, and so on across all the observations in the estimation sample. This is the correct order for recursive models where the likelihood of an observation depends on previously observed (lagged) values, as in AR or ARCH models.

You can change the order of evaluation so EViews evaluates the specification *by equation,* so *the first assignment statement* is evaluated *for all the observations*, then the second assignment statement is evaluated for all the observations, and so on for each of the assignment statements in the specification. This is the correct order for models where aggregate statistics from intermediate series are used as input to subsequent calculations.

You can explicitly select which method of evaluation you would like by adding a statement to the likelihood specification. To force evaluation by equation, simply add a line containing the keyword "@byeqn". To explicitly state that you require evaluation by observation, the "@byobs" keyword can be used. If no keyword is provided, `@byobs` is assumed.

In the conditional heteroskedasticity example above, it does not matter whether the assignment statements are evaluated by equation (line by line) or by observation, since the results do not depend upon the order of evaluation.

However, if the specification has a recursive structure, or if the specification requires the calculation of aggregate statistics based on intermediate series, you must select the appropriate evaluation order if the calculations are to be carried out correctly.

As an example of the `@byeqn` statement, consider the following specification:

```
@logl robust1
@byeqn
res1 = y-c(1)-c(2)*x
delta = @abs(res1)/6/@median(@abs(res1))
weight = (delta<1)*(1-delta^2)^2
robust1 = -(weight*res1^2)
```

This specification performs robust regression by downweighting outlier residuals at each iteration. The assignment statement for DELTA computes the median of the absolute value of the residuals in each iteration, and this is used as a reference point for forming a weighting function for outliers. The `@byeqn` statement instructs EViews to compute all residuals RES1 at a given iteration before computing the median of those residuals when calculating the DELTA series.

Analytic Derivatives

By default, when maximizing the likelihood and forming estimates of the standard errors, EViews computes numeric derivatives of the likelihood function with respect to the parameters. If you would like to specify an analytic expression for one or more of the derivatives, you may use the `@deriv` statement. The `@deriv` statement has the form:

```
@deriv pname1 sname1 pname2 sname2 …
```

where `pname` is a parameter in the model and `sname` is the name of the corresponding derivative series generated by the specification.

For example, consider the following likelihood object that specifies a multinomial logit model:

```
' multinomial logit with 3 outcomes
@logl logl1
```

```
xb2 = b2(1)+b2(2)*x1+b2(3)*x2
xb3 = b3(1)+b3(2)*x1+b3(3)*x2
denom = 1+exp(xb2)+exp(xb3)
' derivatives wrt the 2nd outcome params
@deriv b2(1) grad21 b2(2) grad22 b2(3) grad23
grad21 = d2-exp(xb2)/denom
grad22 = grad21*x1
grad23 = grad21*x2
' derivatives wrt the 3rd outcome params
@deriv b3(1) grad31 b3(2) grad32 b3(3) grad33
grad31 = d3-exp(xb3)/denom
grad32 = grad31*x1
grad33 = grad31*x2
' specify log likelihood
logl1 = d2*xb2+d3*xb3-log(1+exp(xb2)+exp(xb3))
```

See Greene (1997), Chapter 19.7.1 for a discussion of multinomial logit models. There are three possible outcomes, and the parameters of the three regressors (X1, X2 and the constant) are normalized relative to the first outcome. The analytic derivatives are particularly simple for the multinomial logit model and the two `@deriv` statements in the specification instruct EViews to use the expressions for GRAD21, GRAD22, GRAD23, GRAD31, GRAD32, and GRAD33, instead of computing numeric derivatives.

When working with analytic derivatives, you may wish to check the validity of your expressions for the derivatives by comparing them with numerically computed derivatives. EViews provides you with tools which will perform this comparison at the current values of parameters or at the specified starting values. See the discussion of the **Check Derivatives** view of the likelihood object in "Check Derivatives" on page 683.

Derivative Step Sizes

If analytic derivatives are not specified for any of your parameters, EViews numerically evaluates the derivatives of the likelihood function for those parameters. The step sizes used in computing the derivatives are controlled by two parameters: r (relative step size) and m (minimum step size). Let $\theta^{(i)}$ denote the value of the parameter θ at iteration i. Then the step size at iteration $i+1$ is determined by:

$$s^{(i+1)} = \max(r\theta^{(i)}, m) \tag{22.5}$$

The two-sided numeric derivative is evaluated as:

$$\frac{f(\theta^{(i)} + s^{(i+1)}) - f(\theta^{(i)} - s^{(i+1)})}{2s^{(i+1)}} \tag{22.6}$$

The one-sided numeric derivative is evaluated as:

$$\frac{f(\theta^{(i)} + s^{(i+1)}) - f(\theta^{(i)})}{s^{(i+1)}} \tag{22.7}$$

where f is the likelihood function. Two-sided derivatives are more accurate, but require roughly twice as many evaluations of the likelihood function and so take about twice as long to evaluate.

The `@derivstep` statement can be used to control the step size and method used to evaluate the derivative at each iteration. The `@derivstep` keyword should be followed by sets of three arguments: the name of the parameter to be set (or the keyword `@all`), the relative step size, and the minimum step size.

The default setting is (approximately):

```
@derivstep(1) @all 1.49e-8 1e-10
```

where "`1`" in the parentheses indicates that one-sided numeric derivatives should be used and `@all` indicates that the following setting applies to all of the parameters. The first number following `@all` is the relative step size and the second number is the minimum step size. The default relative step size is set to the square root of machine epsilon (1.49×10^{-8}) and the minimum step size is set to $m = 10^{-10}$.

The step size can be set separately for each parameter in a single or in multiple `@derivstep` statements. The evaluation method option specified in parentheses is a global option; it cannot be specified separately for each parameter.

For example, if you include the line:

```
@derivstep(2) c(2) 1e-7 1e-10
```

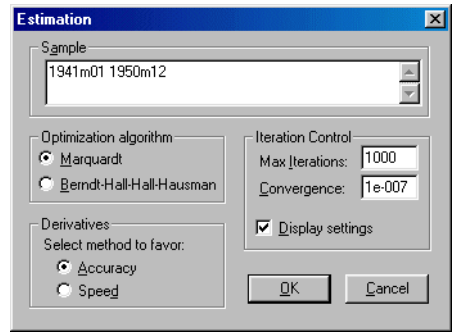
the relative step size for coefficient C(2) will be increased to $m = 10^{-7}$ and a two-sided derivative will be used to evaluate the derivative. In a more complex example,

```
@derivstep(2) @all 1.49e-8 1e-10 c(2) 1e-7 1e-10 c(3) 1e-5 1e-8
```

computes two-sided derivatives using the default step sizes for all coefficients except C(2) and C(3). The values for these latter coefficients are specified directly.

Estimation

Once you have specified the logl object, you can ask EViews to find the parameter values which maximize the likelihood parameters. Simply click the **Estimate** button in the likelihood window toolbar to open the **Estimation Options** dialog.



There are a number of options which allow you to control various aspects of the estimation procedure. See "Setting Estimation Options" on page 951 for a discussion of these options. The default settings, however, should provide a good start for most problems. When you click on **OK**, EViews will begin estimation using the current settings.

Starting Values

Since EViews uses an iterative algorithm to find the maximum likelihood estimates, the choice of starting values is important. For problems in which the likelihood function is globally concave, it will influence how many iterations are taken for estimation to converge. For problems where the likelihood function is not concave, it may determine which of several local maxima is found. In some cases, estimation will fail unless reasonable starting values are provided.

By default, EViews uses the values stored in the coefficient vector or vectors prior to estimation. If a `@param` statement is included in the specification, the values specified in the statement will be used instead.

In our conditional heteroskedasticity regression example, one choice for starting values for the coefficients of the mean equation coefficients are the simple OLS estimates, since OLS provides consistent point estimates even in the presence of (bounded) heteroskedasticity. To use the OLS estimates as starting values, first estimate the OLS equation by the command:

```
equation eq1.ls y c x z
```

After estimating this equation, the elements C(1), C(2), C(3) of the C coefficient vector will contain the OLS estimates. To set the variance scale parameter C(4) to the estimated OLS residual variance, you can type the assignment statement in the command window:

```
c(4) = eq1.@se^2
```

For the final heteroskedasticity parameter C(5), you can use the residuals from the original OLS regression to carry out a second OLS regression, and set the value of C(5) to the appropriate coefficient. Alternatively, you can arbitrarily set the parameter value using a simple assignment statement:

```
c(5) = 1
```

Now, if you estimate the logl specification immediately after carrying out the OLS estimation and subsequent commands, it will use the values that you have placed in the C vector as starting values.

As noted above, an alternative method of initializing the parameters to known values is to include a `@param` statement in the likelihood specification. For example, if you include the line:

```
@param c(1) 0.1 c(2) 0.1 c(3) 0.1 c(4) 1 c(5) 1
```

in the specification of the logl, EViews will always set the starting values to C(1) = C(2) = C(3) = 0.1, C(4) = C(5) = 1.

See also the discussion of starting values in "Starting Coefficient Values" on page 951.

Estimation Sample

EViews uses the sample of observations specified in the Estimation Options dialog when estimating the parameters of the log likelihood. EViews evaluates each expression in the logl for every observation in the sample at current parameter values, using the *by observation* or *by equation* ordering. All of these evaluations follow the standard EViews rules for evaluating series expressions.

If there are missing values in the log likelihood series at the initial parameter values, EViews will issue an error message and the estimation procedure will stop. In contrast to the behavior of other EViews built-in procedures, logl estimation performs no endpoint adjustments or dropping of observations with missing values when estimating the parameters of the model.

LogL Views

- **Likelihood Specification**: displays the window where you specify and edit the likelihood specification.
- **Estimation Output**: displays the estimation results obtained from maximizing the likelihood function.
- **Covariance Matrix**: displays the estimated covariance matrix of the parameter estimates. These are computed from the inverse of the sum of the outer product of the first derivatives evaluated at the optimum parameter values. To save this covariance matrix as a (SYM) MATRIX, you may use the `@cov` function.
- **Wald Coefficient Tests...**: performs the Wald coefficient restriction test. See "Wald Test (Coefficient Restrictions)" on page 572, for a discussion of Wald tests.

- **Gradients**: displays view of the gradients (first derivatives) of the log likelihood at the current parameter values (if the model has not yet been estimated), or at the converged parameter values (if the model has been estimated). These views may prove to be useful diagnostic tools if you are experiencing problems with convergence.
- **Check Derivatives**: displays the values of the numeric derivatives and analytic derivatives (if available) at the starting values (if a `@param` statement is included), or at current parameter values (if there is no `@param` statement).

LogL Procs

- **Estimate...**: brings up a dialog to set estimation options, and to estimate the parameters of the log likelihood.
- **Make Model**: creates an untitled model object out of the estimated likelihood specification.
- **Make Gradient Group**: creates an untitled group of the gradients (first derivatives) of the log likelihood at the estimated parameter values. These gradients are often used in constructing Lagrange multiplier tests.
- **Update Coefs from LogL**: updates the coefficient vector(s) with the estimates from the likelihood object. This procedure allows you to export the maximum likelihood estimates for use as starting values in other estimation problems.

Most of these procedures should be familiar to you from other EViews estimation objects. We describe below the features that are specific to the logl object.

Estimation Output

In addition to the coefficient and standard error estimates, the standard output for the logl object describes the method of estimation, sample used in estimation, date and time that the logl was estimated, evaluation order, and information about the convergence of the estimation procedure.

LogL: MLOGIT
Method: Maximum Likelihood (Marquardt)
Date: 01/16/04 Time: 08:36
Sample: 1 1000
Included observations: 1000
Evaluation order: By observation
Estimation settings: tol= 1.0e-09, derivs=analytic
Initial Values: B2(1)=-1.08356, B2(2)=0.90467, B2(3)=-0.06786, B3(1)=
-0.69842, B3(2)=-0.33212, B3(3)=0.32981
Convergence achieved after 7 iterations

	Coefficient	Std. Error	z-Statistic	Prob.
B2(1)	-0.521793	0.205568	-2.538302	0.0111
B2(2)	0.994358	0.267963	3.710798	0.0002
B2(3)	0.134983	0.265655	0.508115	0.6114
B3(1)	-0.262307	0.207174	-1.266122	0.2055
B3(2)	0.176770	0.274756	0.643371	0.5200
B3(3)	0.399166	0.274056	1.456511	0.1453

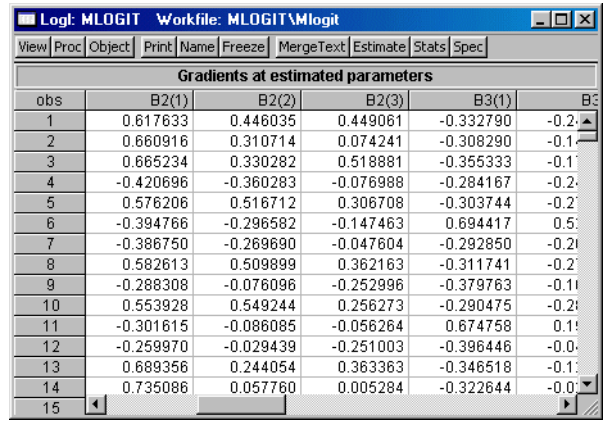
Log likelihood	-1089.415	Akaike info criterion	2.190830
Avg. log likelihood	-1.089415	Schwarz criterion	2.220277
Number of Coefs.	6	Hannan-Quinn criter.	2.202022

EViews also provides the log likelihood value, average log likelihood value, number of coefficients, and three Information Criteria. By default, the starting values are not displayed. Here, we have used the **Estimation Options** dialog to instruct EViews to display the estimation starting values in the output.

Gradients

The gradient summary table and gradient summary graph view allow you to examine the gradients of the likelihood. These gradients are computed at the current parameter values (if the model has not yet been estimated), or at the converged parameter values (if the model has been estimated). See Appendix D, "Gradients and Derivatives", on page 963 for additional details.

You may find this view to be a useful diagnostic tool when experiencing problems with convergence or singularity. One common problem leading to singular matrices is a zero derivative for a parameter due to an incorrectly specified likelihood, poor starting values, or a lack of model identification. See the discussion below for further details.

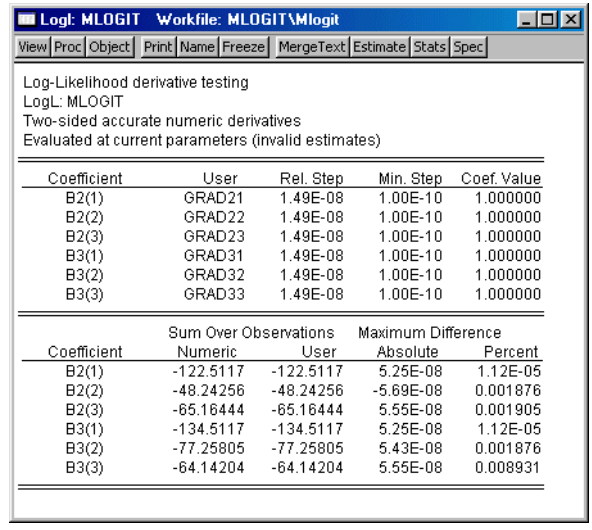

Logl: MLOGIT Workfile: MLOGIT\Mlogit
View Proc Object Print Name Freeze MergeText Estimate Stats Spec
Gradients at estimated parameters

obs	B2(1)	B2(2)	B2(3)	B3(1)	B3
1	0.617633	0.446035	0.449061	-0.332790	-0.2
2	0.660916	0.310714	0.074241	-0.308290	-0.1
3	0.665234	0.330282	0.518881	-0.355333	-0.1
4	-0.420696	-0.360283	-0.076988	-0.284167	-0.2
5	0.576206	0.516712	0.306708	-0.303744	-0.2
6	-0.394766	-0.296582	-0.147463	0.694417	0.5
7	-0.386750	-0.269690	-0.047604	-0.292850	-0.2
8	0.582613	0.509899	0.362163	-0.311741	-0.2
9	-0.288308	-0.076096	-0.252996	-0.379763	-0.1
10	0.553928	0.549244	0.256273	-0.290475	-0.2
11	-0.301615	-0.086085	-0.056264	0.674758	0.1
12	-0.259970	-0.029439	-0.251003	-0.396446	-0.0
13	0.689356	0.244054	0.363363	-0.346518	-0.1
14	0.735086	0.057760	0.005284	-0.322644	-0.0
15					

Check Derivatives

You can use the **Check Derivatives** view to examine your numeric derivatives or to check the validity of your expressions for the analytic derivatives. If the logl specification contains a `@param` statement, the derivatives will be evaluated at the specified values, otherwise, the derivatives will be computed at the current coefficient values.

The first part of this view displays the names of the user supplied derivatives, step size parameters, and the coefficient values at which the derivatives are evaluated. The relative and minimum step sizes shown in this example are the default settings.


Logl: MLOGIT Workfile: MLOGIT\Mlogit
View Proc Object Print Name Freeze MergeText Estimate Stats Spec
Log-Likelihood derivative testing
LogL: MLOGIT
Two-sided accurate numeric derivatives
Evaluated at current parameters (invalid estimates)

Coefficient	User	Rel. Step	Min. Step	Coef. Value
B2(1)	GRAD21	1.49E-08	1.00E-10	1.000000
B2(2)	GRAD22	1.49E-08	1.00E-10	1.000000
B2(3)	GRAD23	1.49E-08	1.00E-10	1.000000
B3(1)	GRAD31	1.49E-08	1.00E-10	1.000000
B3(2)	GRAD32	1.49E-08	1.00E-10	1.000000
B3(3)	GRAD33	1.49E-08	1.00E-10	1.000000

	Sum Over Observations		Maximum Difference	
Coefficient	Numeric	User	Absolute	Percent
B2(1)	-122.5117	-122.5117	5.25E-08	1.12E-05
B2(2)	-48.24256	-48.24256	-5.69E-08	0.001876
B2(3)	-65.16444	-65.16444	5.55E-08	0.001905
B3(1)	-134.5117	-134.5117	5.25E-08	1.12E-05
B3(2)	-77.25805	-77.25805	5.43E-08	0.001876
B3(3)	-64.14204	-64.14204	5.55E-08	0.008931

The second part of the view computes the sum (over all individuals in the sample) of the numeric and, if applicable, the analytic derivatives for each coefficient. If appropriate, EViews will also compute the largest individual difference between the analytic and the numeric derivatives in both absolute, and percentage terms.

Troubleshooting

Because the logl object provides a great deal of flexibility, you are more likely to experience problems with estimation using the logl object than with EViews' built-in estimators.

If you are experiencing difficulties with estimation the following suggestions may help you in solving your problem:

- **Check your likelihood specification**. A simple error involving a wrong sign can easily stop the estimation process from working. You should also verify that the parameters of the model are really identified (in some specifications you may have to impose a normalization across the parameters). Also, every parameter which appears in the model must feed directly or indirectly into the likelihood contributions. The **Check Derivatives** view is particularly useful in helping you spot the latter problem.

- **Choose your starting values**. If any of the likelihood contributions in your sample cannot be evaluated due to missing values or because of domain errors in mathematical operations (logs and square roots of negative numbers, division by zero, etc.) the estimation will stop immediately with the message: "Cannot compute @logl due to missing values". In other cases, a bad choice of starting values may lead you into regions where the likelihood function is poorly behaved. You should always try to initialize your parameters to sensible numerical values. If you have a simpler estimation technique available which approximates the problem, you may wish to use estimates from this method as starting values for the maximum likelihood specification.

- **Make sure lagged values are initialized correctly**. In contrast to most other estimation routines in EViews, the logl estimation procedure will not automatically drop observations with NAs or lags from the sample when estimating a log likelihood model. If your likelihood specification involves lags, you will either have to drop observations from the beginning of your estimation sample, or you will have to carefully code the specification so that missing values from before the sample do not cause NAs to propagate through the entire sample (see the AR(1) and GARCH examples for a demonstration).

Since the series used to evaluate the likelihood are contained in your workfile (unless you use the `@temp` statement to delete them), you can examine the values in the log likelihood and intermediate series to find problems involving lags and missing values.

- **Verify your derivatives**. If you are using analytic derivatives, use the **Check Derivatives** view to make sure you have coded the derivatives correctly. If you are using numerical derivatives, consider specifying analytic derivatives or adjusting the options for derivative method or step size.

- **Reparametrize your model**. If you are having problems with parameter values causing mathematical errors, you may wish to consider reparameterizing the model to restrict the parameter within its valid domain. See the discussion below for examples.

Most of the error messages you are likely to see during estimation are self-explanatory. The error message "near singular matrix" may be less obvious. This error message occurs when EViews is unable to invert the matrix of the sum of the outer product of the derivatives so that it is impossible to determine the direction of the next step of the optimization. This error may indicate a wide variety of problems, including bad starting values, but will almost always occur if the model is not identified, either theoretically, or in terms of the available data.

Limitations

The likelihood object can be used to estimate parameters that maximize (or minimize) a variety of objective functions. Although the main use of the likelihood object will be to specify a log likelihood, you can specify least squares and minimum distance estimation problems with the likelihood object as long as the objective function is additive over the sample.

You should be aware that the algorithm used in estimating the parameters of the log likelihood is not well suited to solving arbitrary maximization or minimization problems. The algorithm forms an approximation to the Hessian of the log likelihood, based on the sum of the outer product of the derivatives of the likelihood contributions. This approximation relies on both the functional form and statistical properties of maximum likelihood objective functions, and may not be a good approximation in general settings. Consequently, you may or may not be able to obtain results with other functional forms. Furthermore, the standard error estimates of the parameter values will only have meaning if the series describing the log likelihood contributions are (up to an additive constant) the individual contributions to a correctly specified, well-defined theoretical log likelihood.

Currently, the expressions used to describe the likelihood contribution must follow the rules of EViews series expressions. This restriction implies that we do not allow matrix operations in the likelihood specification. In order to specify likelihood functions for multiple equation models, you may have to write out the expression for the determinants and quadratic forms. Although possible, this may become tedious for models with more than two or three equations. See the multivariate GARCH sample programs for examples of this approach.

Additionally, the logl object does not directly handle optimization subject to general inequality constraints. There are, however, a variety of well-established techniques for imposing simple inequality constraints. We provide examples below. The underlying idea is to

apply a monotonic transformation to the coefficient so that the new coefficient term takes on values only in the desired range. The commonly used transformations are the `@exp` for one-sided restrictions and the `@logit` and `@arctan` for two-sided restrictions.

You should be aware of the limitations of the transformation approach. First, the approach only works for relatively simple inequality constraints. If you have several cross-coefficient inequality restrictions, the solution will quickly become intractable. Second, in order to perform hypothesis tests on the untransformed coefficient, you will have to obtain an estimate of the standard errors of the associated expressions. Since the transformations are generally nonlinear, you will have to compute linear approximations to the variances yourself (using the delta method). Lastly, inference will be poor near the boundary values of the inequality restrictions.

Simple One-Sided Restrictions

Suppose you would like to restrict the estimate of the coefficient of X to be no larger than 1. One way you could do this is to specify the corresponding subexpression as follows:

```
' restrict coef on x to not exceed 1
res1 = y - c(1) - (1-exp(c(2)))*x
```

Note that EViews will report the point estimate and the standard error for the parameter C(2), not the coefficient of X. To find the standard error of the expression `1-exp(c(2))`, you will have to use the delta method; see for example Greene (1997), Theorems 4.15 and 4.16.

Simple Two-Sided Restrictions

Suppose instead that you want to restrict the coefficient for X to be between -1 and 1. Then you can specify the expression as:

```
' restrict coef on x to be between -1 and 1
res1 = y - c(1) - (2*@logit(c(2))-1)*x
```

Again, EViews will report the point estimate and standard error for the parameter C(2). You will have to use the delta method to compute the standard error of the transformation expression `2*@logit(c(2))-1`.

More generally, if you want to restrict the parameter to lie between L and H, you can use the transformation:

```
(H-L)*@logit(c(1)) + L
```

where C(1) is the parameter to be estimated. In the above example, L = -1 and H = 1.

Examples

In this section, we provide extended examples of working with the logl object to estimate a multinomial logit and a maximum likelihood AR(1) specification. Example programs for these and several other specifications are provided in your default EViews data directory. If you set your default directory to point to the EViews data directory, you should be able to issue a RUN command for each of these programs to create the logl object and to estimate the unknown parameters.

Multinomial Logit (mlogit1.prg)

In this example, we demonstrate how to specify and estimate a simple multinomial logit model using the logl object. Suppose the dependent variable Y can take one of three categories 1, 2, and 3. Further suppose that there are data on two regressors, X1 and X2 that vary across observations (individuals). Standard examples include variables such as age and level of education. Then the multinomial logit model assumes that the probability of observing each category in Y is given by:

$$\Pr(y_i = j) = \frac{\exp(\beta_{0j} + \beta_{1j}x_{1i} + \beta_{2j}x_{2i})}{\sum_{k=1}^{3} \exp(\beta_{0k} + \beta_{1k}x_{1i} + \beta_{2k}x_{2i})} = P_{ij} \tag{22.8}$$

for $j = 1, 2, 3$. Note that the parameters β are specific to each category so there are $3 \times 3 = 9$ parameters in this specification. The parameters are not all identified unless we impose a normalization (see for example Greene, 1997, chapter 19.7), so we normalize the parameters of the first choice category $j = 1$ to be all zero: $\beta_{0,1} = \beta_{1,1} = \beta_{2,1} = 0$.

The log likelihood function for the multinomial logit can be written as:

$$l = \sum_{i=1}^{N} \sum_{j=1}^{3} d_{ij} \log(P_{ij}) \tag{22.9}$$

where d_{ij} is a dummy variable that takes the value 1 if observation i has chosen alternative j and 0 otherwise. The first-order conditions are:

$$\frac{\partial l}{\partial \beta_{kj}} = \sum_{i=1}^{N} (d_{ij} - P_{ij}) x_{ki} \tag{22.10}$$

for $k = 0, 1, 2$ and $j = 1, 2, 3$.

We have provided, in the Example Files subdirectory of your default EViews directory, a workfile MLOGIT.WK1 containing artificial multinomial data. The program begins by loading this workfile:

```
' load artificial data
```

```
%evworkfile = @evpath + "\example files\logl\mlogit"
load "{%evworkfile}"
```

from the EViews example directory.

Next, we declare the coefficient vectors that will contain the estimated parameters for each choice alternative:

```
' declare parameter vector
coef(3) b2
coef(3) b3
```

As an alternative, we could have used the default coefficient vector C.

We then set up the likelihood function by issuing a series of append statements:

```
mlogit.append xb2 = b2(1)+b2(2)*x1+b2(3)*x2
mlogit.append xb3 = b3(1)+b3(2)*x1+b3(3)*x2
' define prob for each choice
mlogit.append denom = 1+exp(xb2)+exp(xb3)
mlogit.append pr1 = 1/denom
mlogit.append pr2 = exp(xb2)/denom
mlogit.append pr3 = exp(xb3)/denom
' specify likelihood
mlogit.append logl1 = (1-dd2-dd3)*log(pr1)
   +dd2*log(pr2)+dd3*log(pr3)
```

Since the analytic derivatives for the multinomial logit are particularly simple, we also specify the expressions for the analytic derivatives to be used during estimation and the appropriate `@deriv` statements:

```
' specify analytic derivatives
for!i = 2 to 3
mlogit.append @deriv b{!i}(1) grad{!i}1 b{!i}(2) grad{!i}2
   b{!i}(3) grad{!i}3
mlogit.append grad{!i}1 = dd{!i}-pr{!i}
mlogit.append grad{!i}2 = grad{!i}1*x1
mlogit.append grad{!i}3 = grad{!i}1*x2
next
```

Note that if you were to specify this likelihood interactively, you would simply type the expression that follows each append statement directly into the MLOGIT object.

This concludes the actual specification of the likelihood object. Before estimating the model, we get the starting values by estimating a series of binary logit models:

```
' get starting values from binomial logit
equation eq2.binary(d=l) dd2 c x1 x2
b2 = eq2.@coefs
equation eq3.binary(d=l) dd3 c x1 x2
b3 = eq3.@coefs
```

To check whether you have specified the analytic derivatives correctly, choose **View/ Check Derivatives** or use the command:

```
show mlogit.checkderiv
```

If you have correctly specified the analytic derivatives, they should be fairly close to the numeric derivatives.

We are now ready to estimate the model. Either click the **Estimate** button or use the command:

```
' do MLE
mlogit.ml(showopts, m=1000, c=1e-5)
show mlogit.output
```

Note that you can examine the derivatives for this model using the **Gradient Table** view, or you can examine the series in the workfile containing the gradients. You can also look at the intermediate results and log likelihood values. For example, to look at the likelihood contributions for each individual, simply double click on the LOGL1 series.

AR(1) Model (ar1.prg)

In this example, we demonstrate how to obtain full maximum likelihood estimates of an AR(1). The maximum likelihood procedure uses the first observation in the sample, in contrast to the built-in AR(1) procedure in EViews which treats the first observation as fixed and maximizes the conditional likelihood for the remaining observations by nonlinear least squares.

As an illustration, we first generate data that follows an AR(1) process:

```
' make up data
create m 80 89
rndseed 123
series y=0
smpl @first+1 @last
y = 1+0.85*y(-1) + nrnd
```

The exact Gaussian likelihood function for an AR(1) model is given by:

$$f(y,\theta) = \begin{cases} \frac{1}{\sigma\sqrt{2\pi(1-\rho^2)}}\exp\left\{-\frac{(y_t - c/(1-\rho^2))^2}{2(\sigma^2/(1-\rho^2))}\right\} & t = 1 \\ \frac{1}{\sigma\sqrt{2\pi}}\exp\left\{-\frac{(y_t - c - \rho y_{t-1})^2}{2(\sigma^2)}\right\} & t > 0 \end{cases} \tag{22.11}$$

where c is the constant term, ρ is the AR(1) coefficient, and σ^2 is the error variance, all to be estimated (see for example Hamilton, 1994a, chapter 5.2).

Since the likelihood function evaluation differs for the first observation in our sample, we create a dummy variable indicator for the first observation:

```
' create dummy variable for first obs
series d1 = 0
smpl @first @first
d1 = 1
smpl @all
```

Next, we declare the coefficient vectors to store the parameter estimates and initialize them with the least squares estimates:

```
' set starting values to LS (drops first obs)
equation eq1.ls y c ar(1)
coef(1) rho = c(2)
coef(1) s2 = eq1.@se^2
```

We then specify the likelihood function. We make use of the `@recode` function to differentiate the evaluation of the likelihood for the first observation from the remaining observations. Note: the `@recode` function used here uses the updated syntax for this function—please double-check the current documentation for details.

```
' set up likelihood
logl ar1
ar1.append @logl logl1
ar1.append var = @recode(d1=1,s2(1)/(1-rho(1)^2),s2(1))
ar1.append res = @recode(d1=1,y-c(1)/(1-rho(1)),y-c(1)-
   rho(1)*y(-1))
ar1.append sres = res/@sqrt(var)
ar1.append logl1 = log(@dnorm(sres))-log(var)/2
```

The likelihood specification uses the built-in function `@dnorm` for the standard normal density. The second term is the Jacobian term that arises from transforming the standard normal variable to one with non-unit variance. (You could, of course, write out the likelihood for the normal distribution without using the `@dnorm` function.)

The program displays the MLE together with the least squares estimates:

```
' do MLE
ar1.ml(showopts, m=1000, c=1e-5)
show ar1.output
' compare with EViews AR(1) which ignores first obs
show eq1.output
```

Additional Examples

The following additional example programs can be found in the "Example Files" subdirectory of your default EViews directory.

- **Conditional logit** (clogit1.prg): estimates a conditional logit with 3 outcomes and both individual specific and choice specific regressors. The program also displays the prediction table and carries out a Hausman test for independence of irrelevant alternatives (IIA). See Greene (1997, chapter 19.7) for a discussion of multinomial logit models.

- **Box-Cox transformation** (boxcox1.prg): estimates a simple bivariate regression with an estimated Box-Cox transformation on both the dependent and independent variables. Box-Cox transformation models are notoriously difficult to estimate and the results are very sensitive to starting values.

- **Disequilibrium switching model** (diseq1.prg): estimates the switching model in exercise 15.14–15.15 of Judge et al. (1985, pages 644–646). Note that there are some typos in Judge et al. (1985, pages 639–640). The program uses the likelihood specification in Quandt (1988, page 32, equations 2.3.16–2.3.17).

- **Multiplicative heteroskedasticity** (hetero1.prg): estimates a linear regression model with multiplicative heteroskedasticity. Replicates the results in Greene (1997, example 12.14).

- **Probit with heteroskedasticit**y (hprobit1.prg): estimates a probit specification with multiplicative heteroskedasticity. See Greene (1997, example 19.7).

- **Probit with grouped data** (gprobit1.prg): estimates a probit with grouped data (proportions data). Estimates the model in Greene (1997, exercise 19.6).

- **Nested logit** (nlogit1.prg): estimates a nested logit model with 2 branches. Tests the IIA assumption by a Wald test. See Greene (1997, chapter 19.7.4) for a discussion of nested logit models.
- **Zero-altered Poisson model** (zpoiss1.prg): estimates the zero-altered Poisson model. Also carries out the non-nested LR test of Vuong (1989). See Greene (1997, chapter 19.9.6) for a discussion of zero-altered Poisson models and Vuong's non-nested likelihood ratio test.
- **Heckman sample selection model** (heckman1.prg): estimates Heckman's two equation sample selection model by MLE using the two-step estimates as starting values.
- **Weibull hazard model** (weibull1.prg): estimates the uncensored Weibull hazard model described in Greene (1997, example 20.18). The program also carries out one of the conditional moment tests in Greene (1997, example 20.19).
- **GARCH(1,1) with t-distributed errors** (arch_t1.prg): estimates a GARCH(1,1) model with *t*-distribution. The log likelihood function for this model can be found in Hamilton (1994a, equation 21.1.24, page 662). Note that this model may more easily be estimated using the standard ARCH estimation tools provided in EViews.
- **GARCH with coefficient restrictions** (garch1.prg): estimates an MA(1)-GARCH(1,1) model with coefficient restrictions in the conditional variance equation. This model is estimated by Bollerslev, Engle, and Nelson (1994, equation 9.1, page 3015) for different data.
- **EGARCH with generalized error distributed errors** (egarch1.prg): estimates Nelson's (1991) exponential GARCH with generalized error distribution. The specification and likelihood are described in Hamilton (1994a, pages 668–669). Note that this model may more easily be estimated using the standard ARCH estimation tools provided in EViews (Chapter 20, "ARCH and GARCH Estimation", on page 601).
- **Multivariate GARCH** (bv_garch.prg and tv_garch.prg): estimates the bi- or the trivariate version of the BEKK GARCH specification (Engle and Kroner, 1995).

Part V. Multiple Equation Analysis

In this section, we document EViews tools for multiple equation estimation, forecasting and data analysis.

- The first two chapters describe estimation techniques for systems of equations (Chapter 23, "System Estimation", on page 695), and VARs and VECs (Chapter 24, "Vector Autoregression and Error Correction Models", on page 721).
- Chapter 25, "State Space Models and the Kalman Filter", on page 753 describes the use of EViews' state space and Kalman filter tools for modeling structural time series models.
- Chapter 26, "Models", beginning on page 777 describes the use of model objects to forecast from multiple equation estimates, or to perform multivariate simulation.

Chapter 23. System Estimation

This chapter describes methods of estimating the parameters of systems of equations. We describe least squares, weighted least squares, seemingly unrelated regression (SUR), weighted two-stage least squares, three-stage least squares, full-information maximum likelihood (FIML), and generalized method of moments (GMM) estimation techniques.

Once you have estimated the parameters of your system of equations, you may wish to forecast future values or perform simulations for different values of the explanatory variables. Chapter 26, "Models", on page 777 describes the use of models to forecast from an estimated system of equations or to perform single and multivariate simulation.

Background

A *system* is a group of equations containing unknown parameters. Systems can be estimated using a number of multivariate techniques that take into account the interdependencies among the equations in the system.

The general form of a system is:

$$f(y_t, x_t, \beta) = \epsilon_t, \tag{23.1}$$

where y_t is a vector of endogenous variables, x_t is a vector of exogenous variables, and ϵ_t is a vector of possibly serially correlated disturbances. The task of estimation is to find estimates of the vector of parameters β.

EViews provides you with a number of methods of estimating the parameters of the system. One approach is to estimate each equation in the system separately, using one of the single equation methods described earlier in this manual. A second approach is to estimate, simultaneously, the complete set of parameters of the equations in the system. The simultaneous approach allows you to place constraints on coefficients across equations and to employ techniques that account for correlation in the residuals across equations.

While there are important advantages to using a system to estimate your parameters, they do not come without cost. Most importantly, if you misspecify one of the equations in the system and estimate your parameters using single equation methods, only the misspecified equation will be poorly estimated. If you employ system estimation techniques, the poor estimates for the misspecification equation may "contaminate" estimates for other equations.

At this point, we take care to distinguish between systems of equations and models. A *model* is a group of known equations describing endogenous variables. Models are used to solve for values of the endogenous variables, given information on other variables in the model.

Systems and models often work together quite closely. You might estimate the parameters of a system of equations, and then create a model in order to forecast or simulate values of the endogenous variables in the system. We discuss this process in greater detail in Chapter 26, “Models”, on page 777.

System Estimation Methods

EViews will estimate the parameters of a system of equations using:

- Ordinary least squares.
- Equation weighted regression.
- Seemingly unrelated regression (SUR).
- System two-state least squares.
- Weighted two-stage least squares.
- Three-stage least squares.
- Full information maximum likelihood (FIML).
- Generalized method of moments (GMM).

The equations in the system may be linear or nonlinear, and may contain autoregressive error terms.

In the remainder of this section, we describe each technique at a general level. Users who are interested in the technical details are referred to the “Technical Discussion” on page 712.

Ordinary Least Squares

This technique minimizes the sum-of-squared residuals for each equation, accounting for any cross-equation restrictions on the parameters of the system. If there are no such restrictions, this method is identical to estimating each equation using single-equation ordinary least squares.

Cross-Equation Weighting

This method accounts for cross-equation heteroskedasticity by minimizing the weighted sum-of-squared residuals. The equation weights are the inverses of the estimated equation variances, and are derived from unweighted estimation of the parameters of the system. This method yields identical results to unweighted single-equation least squares if there are no cross-equation restrictions.

Seemingly Unrelated Regression

The seemingly unrelated regression (SUR) method, also known as the multivariate regression, or Zellner's method, estimates the parameters of the system, accounting for heteroskedasticity and contemporaneous correlation in the errors across equations. The estimates of the cross-equation covariance matrix are based upon parameter estimates of the unweighted system.

Note that EViews estimates a more general form of SUR than is typically described in the literature, since it allows for cross-equation restrictions on parameters.

Two-Stage Least Squares

The system two-stage least squares (STSLS) estimator is the system version of the single equation two-stage least squares estimator described above. STSLS is an appropriate technique when some of the right-hand side variables are correlated with the error terms, and there is neither heteroskedasticity, nor contemporaneous correlation in the residuals. EViews estimates STSLS by applying TSLS equation by equation to the unweighted system, enforcing any cross-equation parameter restrictions. If there are no cross-equation restrictions, the results will be identical to unweighted single-equation TSLS.

Weighted Two-Stage Least Squares

The weighted two-stage least squares (WTSLS) estimator is the two-stage version of the weighted least squares estimator. WTSLS is an appropriate technique when some of the right-hand side variables are correlated with the error terms, and there is heteroskedasticity, but no contemporaneous correlation in the residuals.

EViews first applies STSLS to the unweighted system. The results from this estimation are used to form the equation weights, based upon the estimated equation variances. If there are no cross-equation restrictions, these first-stage results will be identical to unweighted single-equation TSLS.

Three-Stage Least Squares

Three-stage least squares (3SLS) is the two-stage least squares version of the SUR method. It is an appropriate technique when right-hand side variables are correlated with the error terms, and there is both heteroskedasticity, and contemporaneous correlation in the residuals.

EViews applies TSLS to the unweighted system, enforcing any cross-equation parameter restrictions. These estimates are used to form an estimate of the full cross-equation covariance matrix which, in turn, is used to transform the equations to eliminate the cross-equation correlation. TSLS is applied to the transformed model.

Full Information Maximum Likelihood (FIML)

Full Information Maximum Likelihood (FIML) estimates the likelihood function under the assumption that the contemporaneous errors have a joint normal distribution. Provided that the likelihood function is correctly specified, FIML is fully efficient.

Generalized Method of Moments (GMM)

The GMM estimator belongs to a class of estimators known as M-estimators that are defined by minimizing some criterion function. GMM is a robust estimator in that it does not require information of the exact distribution of the disturbances.

GMM estimation is based upon the assumption that the disturbances in the equations are uncorrelated with a set of instrumental variables. The GMM estimator selects parameter estimates so that the correlations between the instruments and disturbances are as close to zero as possible, as defined by a criterion function. By choosing the weighting matrix in the criterion function appropriately, GMM can be made robust to heteroskedasticity and/or autocorrelation of unknown form.

Many standard estimators, including all of the system estimators provided in EViews, can be set up as special cases of GMM. For example, the ordinary least squares estimator can be viewed as a GMM estimator, based upon the conditions that each of the right-hand side variables is uncorrelated with the residual.

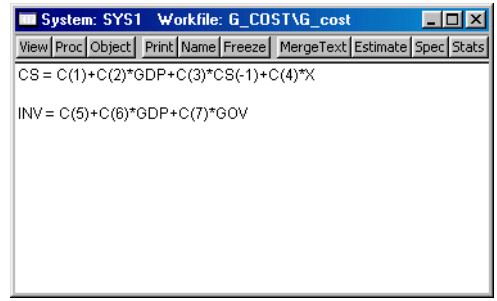
How to Create and Specify a System

To estimate the parameters of your system of equations, you should first create a system object and specify the system of equations. Click on **Object/New Object.../System** or type `system` in the command window. The system object window should appear. When you first create the system, the window will be blank. You will fill the system specification window with text describing the equations, and potentially, lines describing the instruments and the parameter starting values.

Equations

Enter your equations, by formula, using standard EViews expressions. The equations in your system should be behavioral equations with unknown coefficients and an implicit error term.

Consider the specification of a simple two equation system. You can use the default EViews coefficients, C(1), C(2), and so on, or you can use other coefficient vectors, in which case you should first declare them by clicking **Object/New Object.../Matrix-Vector-Coef/Coefficient Vector** in the main menu.


System: SYS1 Workfile: G_COST\G_cost
View Proc Object Print Name Freeze MergeText Estimate Spec Stats
CS = C(1)+C(2)*GDP+C(3)*CS(-1)+C(4)*X
INV = C(5)+C(6)*GDP+C(7)*GOV

There are some general rules for specifying your equations:

- Equations can be nonlinear in their variables, coefficients, or both. Cross equation coefficient restrictions may be imposed by using the same coefficients in different equations. For example:

```
y = c(1) + c(2)*x
z = c(3) + c(2)*z + (1-c(2))*x
```

- You may also impose adding up constraints. Suppose for the equation:

```
y = c(1)*x1 + c(2)*x2 + c(3)*x3
```

  you wish to impose C(1) + C(2) + C(3) = 1. You can impose this restriction by specifying the equation as

```
y = c(1)*x1 + c(2)*x2 + (1-c(1)-c(2))*x3
```

- The equations in a system may contain autoregressive (AR) error specifications, but not MA, SAR, or SMA error specifications. You must associate coefficients with each AR specification. Enclose the entire AR specification in square brackets and follow each AR with an " = "-sign and a coefficient. For example:

```
cs = c(1) + c(2)*gdp + [ar(1)=c(3), ar(2)=c(4)]
```

  You can constrain all of the equations in a system to have the same AR coefficient by giving all equations the same AR coefficient number, or you can estimate separate AR processes, by assigning each equation its own coefficient.

- Equations in a system need not have a dependent variable followed by an equal sign and then an expression. The " = "-sign can be anywhere in the formula, as in:

```
log(unemp/(1-unemp)) = c(1) + c(2)*dmr
```

  You can also write the equation as a simple expression without a dependent variable, as in:

```
(c(1)*x + c(2)*y + 4)^2
```

When encountering an expression that does not contain an equal sign, EViews sets the entire expression equal to the implicit error term.

If an equation should not have a disturbance, it is an identity, and should not be included in a system. If necessary, you should solve out for any identities to obtain the behavioral equations.

You should make certain that there is no identity linking all of the disturbances in your system. For example, if each of your equations describes a fraction of a total, the sum of the equations will always equal one, and the sum of the disturbances will identically equal zero. You will need to drop one of these equations to avoid numerical problems.

Instruments

If you plan to estimate your system using two-stage least squares, three-stage least squares, or GMM, you must specify the instrumental variables to be used in estimation. There are several ways to specify your instruments, with the appropriate form depending on whether you wish to have identical instruments in each equation, and whether you wish to compute the projections on an equation-by-equation basis, or whether you wish to compute a restricted projection using the stacked system.

In the simplest (default) case, EViews will form your instrumental variable projections on an equation-by-equation basis. If you prefer to think of this process as a two-step (2SLS) procedure, the first-stage regression of the variables in your model on the instruments will be run separately for each equation.

In this setting, there are two ways to specify your instruments. If you would like to use identical instruments in every equations, you should include a line beginning with the keyword "@INST" or "INST", followed by a list of all the exogenous variables to be used as instruments. For example, the line:

```
@inst gdp(-1 to -4) x gov
```

instructs EViews to use these six variables as instruments for all of the equations in the system. System estimation will involve a separate projection for each equation in your system.

You may also specify different instruments for each equation by appending an "*@*"-sign at the end of the equation, followed by a list of instruments for that equation. For example:

```
cs = c(1)+c(2)*gdp+c(3)*cs(-1) @ cs(-1) inv(-1) gov
inv = c(4)+c(5)*gdp+c(6)*gov @ gdp(-1) gov
```

The first equation uses CS(-1), INV(-1), GOV, and a constant as instruments, while the second equation uses GDP(-1), GOV, and a constant as instruments.

Lastly, you can mix the two methods. Any equation without individually specified instruments will use the instruments specified by the `@inst` statement. The system:

```
@inst gdp(-1 to -4) x gov
cs = c(1)+c(2)*gdp+c(3)*cs(-1)
inv = c(4)+c(5)*gdp+c(6)*gov @ gdp(-1) gov
```

will use the instruments GDP(-1), GDP(-2), GDP(-3), GDP(-4), X, GOV, and C, for the CS equation, but only GDP(-1), GOV, and C, for the INV equation.

As noted above, the EViews default behavior is to perform the instrumental variables projection on an equation-by-equation basis. You may, however, wish to perform the projections on the stacked system. Notably, where the number of instruments is large, relative to the number of observations, stacking the equations and instruments prior to performing the projection may be the only feasible way to compute 2SLS estimates.

To designate instruments for a stacked projection, you should use the `@stackinst` statement (note: this statement is only available for systems estimated by 2SLS or 3SLS; it is not available for systems estimated using GMM).

In a `@stackinst` statement, the "@STACKINST" keyword should be followed by a list of stacked instrument specifications. Each specification is a comma delimited list of series enclosed in parentheses (one per equation), describing the instruments to be constrained in a stacked specification.

For example, the following `@stackinst` specification creates two instruments in a three equation model:

```
@stackinst (z1,z2,z3) (m1,m1,m1)
```

This statement instructs EViews to form two stacked instruments, one by stacking the separate series Z1, Z2, and Z3, and the other formed by stacking M1 three times. The first-stage instrumental variables projection is then of the variables in the stacked system on the stacked instruments.

When working with systems that have a large number of equations, the above syntax may be unwieldy. For these cases, EViews provides a couple of shortcuts. First, for instruments that are identical in all equations, you may us an "*" after the comma to instruct EViews to repeat the specified series. Thus, the above statement is equivalent to:

```
@stackinst (z1,z2,z3) (m1,*)
```

Second, for non-identical instruments, you may specify a set of stacked instruments using an EViews group object, so long as the number of variables in the group is equal to the number of equations in the system. Thus, if you create a group Z with,

```
group z z1 z2 z3
```

the above statement can be simplified to:

```
@stackinst z (m1,*)
```

You can, of course, combine ordinary instrument and stacked instrument specifications. This situation is equivalent to having common and equation specific coefficients for variables in your system. Simply think of the stacked instruments as representing common (coefficient) instruments, and ordinary instruments as representing equation specific (coefficient) instruments. For example, consider the system given by:

```
@stackinst (z1,z2,z3) (m1,*)
@inst ia
y1 = c(1)*x1
y2 = c(1)*x2
y3 = c(1)*x3 @ ic
```

The stacked instruments for this specification may be represented as:

$$\begin{bmatrix} Z1 & M1 & IA & C & 0 & 0 & 0 & 0 & 0 \\ Z2 & M1 & 0 & 0 & IA & C & 0 & 0 & 0 \\ Z3 & M1 & 0 & 0 & 0 & 0 & IA & C & IC \end{bmatrix} \tag{23.2}$$

so it is easy to see that this specification is equivalent to the following stacked specification,

```
@stackinst (z1, z2, z3) (m1, *) (ia, 0, 0) (0, ia, 0) (0, 0, ia)
    (0, 0, ic)
```

since the common instrument specification,

```
@inst ia
```

is equivalent to:

```
@stackinst (ia, 0, 0) (0, ia, 0) (0, 0, ia)
```

Note that the constant instruments are added implicitly.

Additional Comments

- If you include a "C" in the stacked instrument list, it will not be included in the individual equations. If you do not include the "C" as a stacked instrument, it will be included as an instrument in every equation, whether specified explicitly or not.
- You should list all exogenous right-hand side variables as instruments for a given equation.

- Identification requires that there should be at least as many instruments (including the constant) in each equation as there are right-hand side variables in that equation.
- The `@stackinst` statement is only available for estimation by 2SLS and 3SLS. It is not currently supported for GMM.
- If you estimate your system using a method that does not use instruments, all instrument specification lines will be ignored.

Starting Values

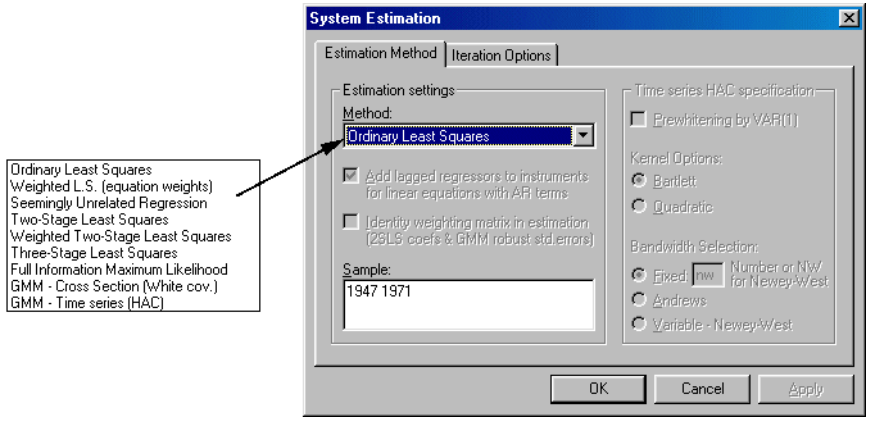
For systems that contain nonlinear equations, you can include a line that begins with `param` to provide starting values for some or all of the parameters. List pairs of parameters and values. For example:

```
param  c(1) .15  b(3) .5
```

sets the initial values of C(1) and B(3). If you do not provide starting values, EViews uses the values in the current coefficient vector.

How to Estimate a System

Once you have created and specified your system, you may push the **Estimate** button on the toolbar to bring up the **System Estimation** dialog.



The combo box marked **Method** provides you with several options for the estimation method. You may choose from one of a number of methods for estimating the parameters of your specification.

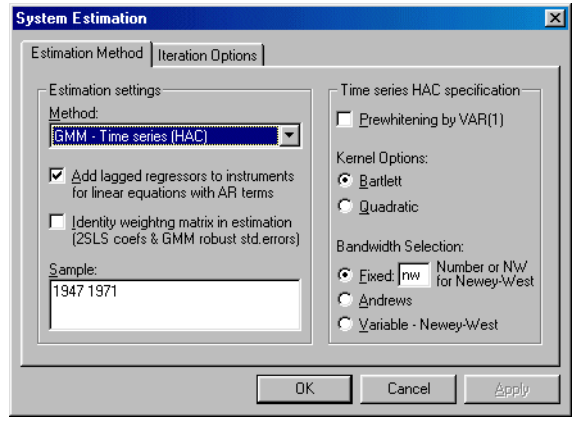
The estimation dialog may change to reflect your choice, providing you with additional options. If you select an estimator which uses instrumental variables, a checkbox will

appear, prompting you to choose whether to **Add lagged regressors to instruments for linear equations with AR terms**. As the checkbox label suggests, if selected, EViews will add lagged values of the dependent and independent variable to the instrument list when estimating AR models. The lag order for these instruments will match the AR order of the specification. This automatic lag inclusion reflects the fact that EViews transforms the linear specification to a nonlinear specification when estimating AR models, and that the lagged values are ideal instruments for the transformed specification. If you wish to maintain precise control over the instruments added to your model, you should unselect this option.

The remaining options appear if you are estimating a GMM specification. Note that the **GMM-Cross section** option uses a weighting matrix that is robust to heteroskedasticity and contemporaneous correlation of unknown form, while the **GMM-Time series (HAC)** option extends this robustness to autocorrelation of unknown form.

If you select either GMM method, EViews will display a checkbox labeled **Identity weighting matrix in estimation**. If selected, EViews will estimate the model using identity weights, and will use the estimated coefficients and GMM specification you provide to compute a coefficient covariance matrix that is robust to cross-section heteroskedasticity (White) or heteroskedasticity and autocorrelation (Newey-West). If this option is not selected, EViews will use the GMM weights both in estimation, and in computing the coefficient covariances.

When you select the **GMM-Time series (HAC)** option, the dialog displays additional options for specifying the weighting matrix. The new options will appear at the lower right part of the dialog control. These options control the computation of the heteroskedasticity and autocorrelation robust (HAC) weighting matrix. See "Technical Discussion" on page 712 for a more detailed discussion of these options.

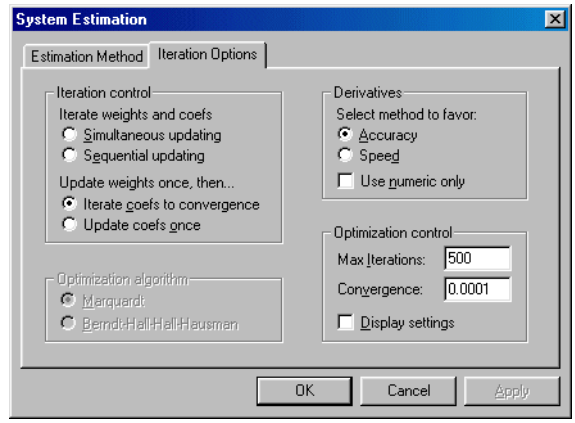


The **Kernel Options** determines the functional form of the kernel used to weight the autocovariances to compute the weighting matrix. The **Bandwidth Selection** option determines how the weights given by the kernel change with the lags of the autocovariances in the computation of the weighting matrix. If you select **Fixed** bandwidth, you may enter a number for the bandwidth or type `nw` to use Newey and West's fixed bandwidth selection criterion.

The **Prewhitening** option runs a preliminary VAR(1) prior to estimation to "soak up" the correlation in the moment conditions.

Iteration Options

For weighted least squares, SUR, weighted TSLS, 3SLS, GMM, and nonlinear systems of equations, there are additional issues involving the procedure for computing the GLS weighting matrix and the coefficient vector. To specify the method used in iteration, click on the **Iteration Option**s tab.



The estimation option controls the method of iterating over coefficients, over the weighting matrices, or both:

- **Update weights once then—Iterate coefs to convergence** is the default method.

  By default, EViews carries out a first-stage estimation of the coefficients using no weighting matrix (the identity matrix). Using starting values obtained from OLS (or TSLS, if there are instruments), EViews iterates the first-stage estimates until the coefficients converge. If the specification is linear, this procedure involves a single OLS or TSLS regression.

  The residuals from this first-stage iteration are used to form a consistent estimate of the weighting matrix.

  In the second stage of the procedure, EViews uses the estimated weighting matrix in forming new estimates of the coefficients. If the model is nonlinear, EViews iterates the coefficient estimates until convergence.

- **Update weights once then—Update coefs once** performs the first-stage estimation of the coefficients, and constructs an estimate of the weighting matrix. In the second stage, EViews does not iterate the coefficients to convergence, instead performing a single coefficient iteration step. Since the first stage coefficients are consistent, this one-step update is asymptotically efficient, but unless the specification is linear, does not produce results that are identical to the first method.

- **Iterate Weights and Coefs—Simultaneous updating** updates both the coefficients and the weighting matrix at each iteration. These steps are then repeated until both

both the coefficients and weighting matrix converge. This is the iteration method employed in EViews prior to version 4.

- **Iterate Weights and Coefs—Sequential updating** repeats the default method of updating weights and then iterating coefficients to convergence until both the coefficients and the weighting matrix converge.

Note that all four of the estimation techniques yield results that are asymptotically efficient. For linear models, the two **Iterate Weights and Coefs** options are equivalent, and the two **One-Step Weighting Matrix** options are equivalent, since obtaining coefficient estimates does not require iteration.

In addition, the **Iteration Options** tab allows you to set a number of options for estimation, including convergence criterion, maximum number of iterations, and derivative calculation settings. See "Setting Estimation Options" on page 951 for related discussion.

Estimation Output

The system estimation output contains parameter estimates, standard errors, and *t*-statistics for each of the coefficients in the system. Additionally, EViews reports the determinant of the residual covariance matrix, and, for FIML estimates, the maximized likelihood value.

In addition, EViews reports a set of summary statistics for each equation. The R^2 statistic, Durbin-Watson statistic, standard error of the regression, sum-of-squared residuals, etc., are computed for each equation using the standard definitions, based on the residuals from the system estimation procedure.

You may access most of these results using regression statistics functions. See Chapter 15, page 454 for a discussion of the use of these functions, and Appendix A, "Object, View and Procedure Reference", on page 153 of the *Command and Programming Reference* for a full listing of the available functions for systems.

Working With Systems

After obtaining estimates, the system object provides a number of tools for examining the equation results, and performing inference and specification testing.

System Views

Some of the system views are familiar from the discussion in previous chapters:

- You can examine the estimated covariance matrix by selecting the **Coefficient Covariance Matrix** view.

- **Wald Coefficient Tests…** performs hypothesis tests on the coefficients. These views are discussed in greater depth in "Wald Test (Coefficient Restrictions)" on page 572.
- The **Estimation Output** view displays the coefficient estimates and summary statistics for the system. You may also access this view by pressing **Stats** on the system toolbar.

Other views are very familiar, but differ slightly in name or output, from their single equation counterparts:

- **System Specification** displays the specification window for the system. The specification window may also be displayed by pressing **Spec** on the toolbar.
- **Residual Graphs** displays a separate graph of the residuals from each equation in the system.
- **Endogenous Table** presents a spreadsheet view of the endogenous variables in the system.
- **Endogenous Graph** displays graphs of each of the endogenous variables.

The last two views are specific to systems:

- **Residual Correlation Matrix** computes the contemporaneous correlation matrix for the residuals of each equation.
- **Residual Covariance Matrix** computes the contemporaneous covariance matrix for the residuals. See also the function `@residcova` in Appendix A, "Object, View and Procedure Reference", on page 184 of the *Command and Programming Reference*.

System Procs

One notable difference between systems and single equation objects is that there is no forecast procedure for systems. To forecast or perform simulation using an estimated system, you must use a model object.

EViews provides you with a simple method of incorporating the results of a system into a model. If you select **Proc/Make Model**, EViews will open an untitled model object containing the estimated system. This model can be used for forecasting and simulation. An alternative approach, creating the model and including the system object by name, is described in "Building a Model" on page 794.

There are other procedures for working with the system:

- **Estimate…** opens the dialog for estimating the system of equations. It may also be accessed by pressing **Estimate** on the system toolbar.

- **Make Residuals** creates a number of series containing the residuals for each equation in the system. The residuals will be given the next unused name of the form RESID01, RESID02, etc., in the order that the equations are specified in the system.
- **Make Endogenous Group** creates an untitled group object containing the endogenous variables.

Example

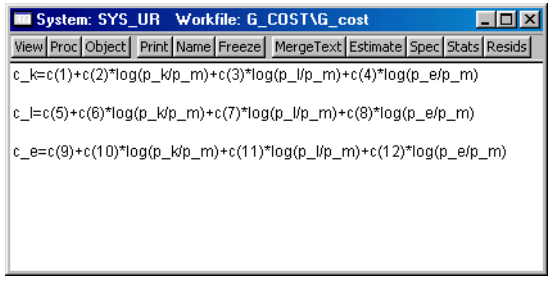
As an illustration of the process of estimating a system of equations in EViews, we estimate a translog cost function using data from Berndt and Wood (1975) as tabulated in Greene (1997). The translog cost function has four factors with three equations of the form:

$$\begin{aligned} c_K &= \beta_K + \delta_{KK}\log\left(\frac{p_K}{p_M}\right) + \delta_{KL}\log\left(\frac{p_L}{p_M}\right) + \delta_{KE}\log\left(\frac{p_E}{p_M}\right) + \epsilon_K \\ c_L &= \beta_L + \delta_{LK}\log\left(\frac{p_K}{p_M}\right) + \delta_{LL}\log\left(\frac{p_L}{p_M}\right) + \delta_{LE}\log\left(\frac{p_E}{p_M}\right) + \epsilon_L \\ c_E &= \beta_E + \delta_{EK}\log\left(\frac{p_K}{p_M}\right) + \delta_{EL}\log\left(\frac{p_L}{p_M}\right) + \delta_{EE}\log\left(\frac{p_E}{p_M}\right) + \epsilon_E \end{aligned} \tag{23.3}$$

where c_i and p_i are the cost share and price of factor i, respectively. β and δ are the parameters to be estimated. Note that there are cross equation coefficient restrictions that ensure symmetry of the cross partial derivatives.

We first estimate this system without imposing the cross equation restrictions and test whether the symmetry restrictions hold. Create a system by clicking **Object/New Object.../System** in the main toolbar or type `system` in the command window. Press the **Name** button and type in the name "SYS_UR" to name the system.

Next, type in the system window and specify the system as:



We estimate this model by full information maximum likelihood (FIML). FIML is invariant to the equation that is dropped. Press the **Estimate** button and choose **Full Information Maximum Likelihood**. EViews presents the estimated coefficients and regression statistics for each equation. The top portion of the output describes the coefficient estimates:

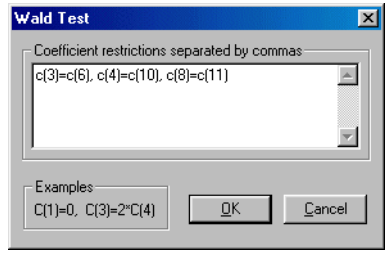
System: SYS_UR
Estimation Method: Full Information Maximum Likelihood (Marquardt)
Date: 01/16/04 Time: 10:15
Sample: 1947 1971
Included observations: 25
Total system (balanced) observations 75
Convergence achieved after 125 iterations

	Coefficient	Std. Error	z-Statistic	Prob.
C(1)	0.054983	0.009352	5.879263	0.0000
C(2)	0.035131	0.035676	0.984709	0.3248
C(3)	0.004134	0.025614	0.161417	0.8718
C(4)	0.023631	0.084439	0.279858	0.7796
C(5)	0.250180	0.012017	20.81915	0.0000
C(6)	0.014758	0.024772	0.595756	0.5513
C(7)	0.083908	0.032184	2.607154	0.0091
C(8)	0.056410	0.096008	0.587548	0.5568
C(9)	0.043257	0.007981	5.420390	0.0000
C(10)	-0.007707	0.012518	-0.615711	0.5381
C(11)	-0.002184	0.020121	-0.108520	0.9136
C(12)	0.035623	0.061801	0.576416	0.5643

Log Likelihood	349.0326
Determinant residual covariance	1.50E-16

while the bottom describes equation specific statistics.

To test the symmetry restrictions, select **View/Wald Coefficient Tests…**, fill in the dialog:



and click **OK**. The result:

Wald Test:
System: SYS_UR

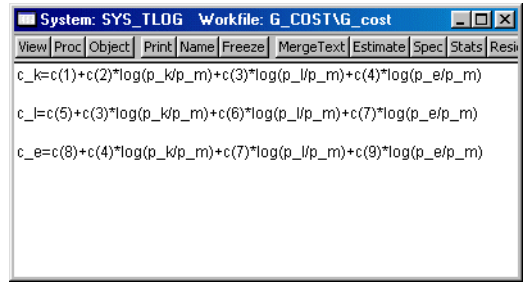
Test Statistic	Value	df	Probability
Chi-square	0.418853	3	0.9363

Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
C(3) - C(6)	-0.010623	0.039839
C(4) - C(10)	0.031338	0.077778
C(8) - C(11)	0.058593	0.090751

Restrictions are linear in coefficients.

fails to reject the symmetry restrictions. To estimate the system imposing the symmetry restrictions, copy the object using **Object/Copy Object,** click **View/System Specification** and modify the system.



We have named the system SYS_TLOG. Note that to impose symmetry in the translog specification, we have restricted the coefficients on the cross-price terms to be the same (we have also renumbered the 9 remaining coefficients so that they are consecutive). The restrictions are imposed by using the same coefficients in each equation. For example, the coefficient on the log(P_L/P_M) term in the C_K equation, C(3), is the same as the coefficient on the log(P_K/P_M) term in the C_L equation.

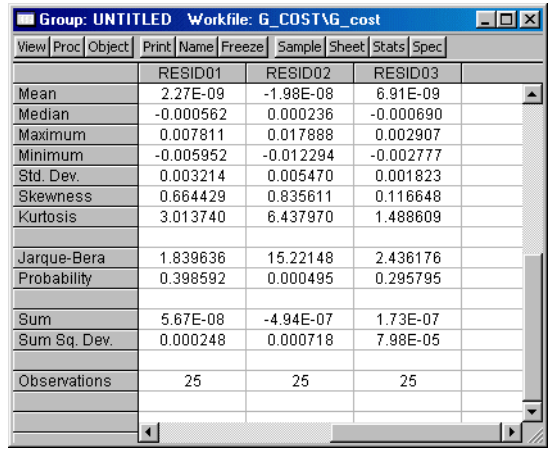
To estimate this model using FIML, click **Estimate** and choose **Full Information Maximum Likelihood**. The top part of the equation describes the estimation specification, and provides coefficient and standard error estimates, t-statistics, p-values, and summary statistics:

System: SYS_TLOG
Estimation Method: Full Information Maximum Likelihood (Marquardt)
Date: 01/16/04 Time: 10:21
Sample: 1947 1971
Included observations: 25
Total system (balanced) observations 75
Convergence achieved after 57 iterations

	Coefficient	Std. Error	z-Statistic	Prob.
C(1)	0.057022	0.003306	17.24910	0.0000
C(2)	0.029742	0.012583	2.363680	0.0181
C(3)	-0.000369	0.011205	-0.032967	0.9737
C(4)	-0.010228	0.006027	-1.697171	0.0897
C(5)	0.253398	0.005050	50.17708	0.0000
C(6)	0.075427	0.015483	4.871573	0.0000
C(7)	-0.004414	0.009141	-0.482901	0.6292
C(8)	0.044286	0.003349	13.22343	0.0000
C(9)	0.018767	0.014894	1.260006	0.2077
Log Likelihood		344.5916		
Determinant residual covariance		2.14E-16		

The log likelihood value reported at the bottom of the first part of the table may be used to construct likelihood ratio tests.

Since maximum likelihood assumes the errors are multivariate normal, we may wish to test whether the residuals are normally distributed. Click **Proc/Make Residuals** and EViews opens an untitled group window containing the residuals of each equation in the system. Then to compute descriptive statistics for each residual in the group, select **View/ Descriptive Stats** from the group window toolbar.



	RESID01	RESID02	RESID03
Mean	2.27E-09	-1.98E-08	6.91E-09
Median	-0.000562	0.000236	-0.000690
Maximum	0.007811	0.017888	0.002907
Minimum	-0.005952	-0.012294	-0.002777
Std. Dev.	0.003214	0.005470	0.001823
Skewness	0.664429	0.835611	0.116648
Kurtosis	3.013740	6.437970	1.488609
Jarque-Bera	1.839636	15.22148	2.436176
Probability	0.398592	0.000495	0.295795
Sum	5.67E-08	-4.94E-07	1.73E-07
Sum Sq. Dev.	0.000248	0.000718	7.98E-05
Observations	25	25	25

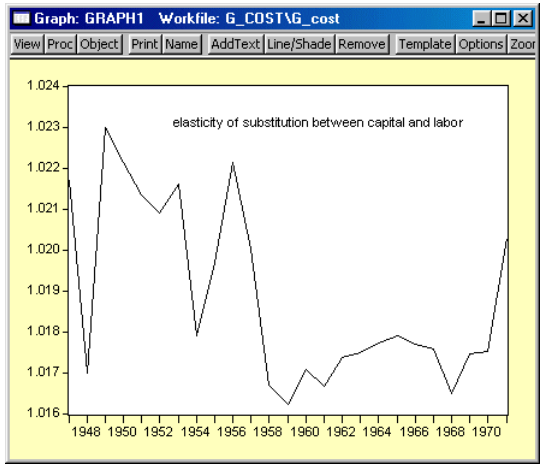
The Jarque-Bera statistic rejects the hypothesis of normal distribution for the second equation but not for the other equations.

The estimated coefficients of the translog cost function may be used to construct estimates of the elasticity of substitution between factors of production. For example, the elasticity of

substitution between capital and labor is given by 1 + c(3)/(C_K*C_L). Note that the elasticity of substitution is not a constant, and depends on the values of C_K and C_L. To create a series containing the elasticities computed for each observation, select **Quick/ Generate Series...**, and enter:

```
es_kl = 1 + sys1.c(3)/(c_k*c_l)
```

To plot the series of elasticity of substitution between capital and labor for each observation, double click on the series name ES_KL in the workfile and select **View/Line Graph**:



While it varies over the sample, the elasticity of substitution is generally close to one, which is consistent with the assumption of a Cobb-Douglas cost function.

Technical Discussion

While the discussion to follow is expressed in terms of a balanced system of linear equations, the analysis carries forward in a straightforward way to unbalanced systems containing nonlinear equations.

Denote a system of m equations in stacked form as:

$$\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_M \end{bmatrix} = \begin{bmatrix} X_1 & 0 & \dots & 0 \\ 0 & X_2 & & \vdots \\ & & \ddots & 0 \\ 0 & \dots & 0 & X_M \end{bmatrix} \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_M \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_M \end{bmatrix} \tag{23.4}$$

where y_m is T vector, X_m is a $T \times k_m$ matrix, and β_m is a k_m vector of coefficients. The error terms ϵ have an $MT \times MT$ covariance matrix V. The system may be written in compact form as:

$$y = X\beta + \epsilon\,. \tag{23.5}$$

Under the standard assumptions, the residual variance matrix from this stacked system is given by:

$$V = \mathrm{E}(\epsilon\epsilon') = \sigma^2(I_M \otimes I_T)\,. \tag{23.6}$$

Other residual structures are of interest. First, the errors may be heteroskedastic across the m equations. Second, they may be heteroskedastic and contemporaneously correlated. We can characterize both of these cases by defining the $M \times M$ matrix of contemporaneous correlations, Σ, where the (i,j)-th element of Σ is given by $\sigma_{ij} = \mathrm{E}(\epsilon_{it}\epsilon_{jt})$ for all t. If the errors are contemporaneously uncorrelated, then, $\sigma_{ij} = 0$ for $i \neq j$, and we can write:

$$V = \mathrm{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_M^2) \otimes I_T \tag{23.7}$$

More generally, if the errors are heteroskedastic and contemporaneously correlated:

$$V = \Sigma \otimes I_T. \tag{23.8}$$

Lastly, at the most general level, there may be heteroskedasticity, contemporaneous correlation, and autocorrelation of the residuals. The general variance matrix of the residuals may be written:

$$V = \begin{pmatrix} \sigma_{11}\Sigma_{11} & \sigma_{12}\Sigma_{12} & \cdots & \sigma_{1M}\Sigma_{1M} \\ \sigma_{21}\Sigma_{21} & \sigma_{22}\Sigma_{22} & & \vdots \\ & & \ddots & \\ \sigma_{M1}\Sigma_{M1} & \cdots & & \sigma_{MM}\Sigma_{MM} \end{pmatrix} \tag{23.9}$$

where Σ_{ij} is an autocorrelation matrix for the i-th and j-th equations.

Ordinary Least Squares

The OLS estimator of the estimated variance matrix of the parameters is valid under the assumption that $V = \Sigma \otimes I_T$. The estimator for β is given by,

$$b_{LS} = (X'X)^{-1}X'y \tag{23.10}$$

and the variance estimator is given by:

$$\mathrm{var}(b_{LS}) = s^2(X'X)^{-1} \tag{23.11}$$

where s^2 is the residual variance estimate for the stacked system.

Weighted Least Squares

The weighted least squares estimator is given by:

$$b_{WLS} = (X'\hat{V}^{-1}X)^{-1}X'\hat{V}^{-1}y \tag{23.12}$$

where $\hat{V} = \text{diag}(s_{11}, s_{22}, \ldots, s_{MM}) \otimes I_T$ is a consistent estimator of V, and s_{ii} is the residual variance estimator,

$$s_{ij} = ((y_i - X_i b_{LS})'(y_j - X_j b_{LS}))/\max(T_i, T_j) \tag{23.13}$$

where the inner product is taken over the non-missing common elements of i and j. The max function in Equation (23.13) is designed to handle the case of unbalanced data by down-weighting the covariance terms. Provided the missing values are asymptotically negligible, this yields a consistent estimator of the variance elements. Note also that there is no adjustment for degrees of freedom.

When specifying your estimation specification, you are given a choice of which coefficients to use in computing the s_{ij}. If you choose not to iterate the weights, the OLS coefficient estimates will be used to estimate the variances. If you choose to iterate the weights, the current parameter estimates (which may be based on the previously computed weights) are used in computing the s_{ij}. This latter procedure may be iterated until the weights and coefficients converge.

The estimator for the coefficient variance matrix is:

$$\text{var}(b_{WLS}) = (X'\hat{V}^{-1}X)^{-1}. \tag{23.14}$$

The weighted least squares estimator is efficient, and the variance estimator consistent, under the assumption that there is heteroskedasticity, but no serial or contemporaneous correlation in the residuals.

It is worth pointing out that if there are no cross-equation restrictions on the parameters of the model, weighted LS on the entire system yields estimates that are identical to those obtained by equation-by-equation LS. Consider the following simple model:

$$\begin{aligned} y_1 &= X_1\beta_1 + \epsilon_1 \\ y_2 &= X_2\beta_2 + \epsilon_2 \end{aligned} \tag{23.15}$$

If β_1 and β_2 are unrestricted, the WLS estimator given in Equation (23.14) yields:

$$b_{WLS} = \begin{bmatrix} ((X_1'X_1)/s_{11})^{-1}((X_1'y_1)/s_{11}) \\ ((X_2'X_2)/s_{22})^{-1}((X_2'y_2)/s_{22}) \end{bmatrix} = \begin{bmatrix} (X_1'X_1)^{-1}X_1'y_1 \\ (X_2'X_2)^{-1}X_2'y_2 \end{bmatrix}. \tag{23.16}$$

The expression on the right is equivalent to equation-by-equation OLS. Note, however, that even without cross-equation restrictions, the standard errors are not the same in the two cases.

Seemingly Unrelated Regression (SUR)

SUR is appropriate when all the right-hand side regressors X are assumed to be exogenous, and the errors are heteroskedastic and contemporaneously correlated so that the error variance matrix is given by $V = \Sigma \otimes I_T$. Zellner's SUR estimator of β takes the form

$$b_{SUR} = (X'(\hat{\Sigma} \otimes I_T)^{-1}X)^{-1}X'(\hat{\Sigma} \otimes I_T)^{-1}y, \tag{23.17}$$

where $\hat{\Sigma}$ is a consistent estimate of Σ with typical element s_{ij}, for all i and j.

If you include AR terms in equation j, EViews transforms the model (see "Estimating AR Models" on page 497) and estimates the following equation:

$$y_{jt} = X_{jt}\beta_j + \left(\sum_{r=1}^{p_j} \rho_{jr}(y_{j(t-r)} - X_{j(t-r)})\right) + \epsilon_{jt} \tag{23.18}$$

where ϵ_j is assumed to be serially independent, but possibly correlated contemporaneously across equations. At the beginning of the first iteration, we estimate the equation by nonlinear LS and use the estimates to compute the residuals $\hat{\epsilon}$. We then construct an estimate of Σ using $s_{ij} = (\hat{\epsilon}_i'\hat{\epsilon}_j)/\max(T_i, T_j)$ and perform nonlinear GLS to complete one iteration of the estimation procedure. These iterations may be repeated until the coefficients and weights converge.

Two-Stage Least Squares (TSLS) and Weighted TSLS

TSLS is a single equation estimation method that is appropriate when some of the variables in X are endogenous. Write the j-th equation of the system as,

$$Y\Gamma_j + XB_j + \epsilon_j = 0 \tag{23.19}$$

or, alternatively:

$$y_j = Y_j\gamma_j + X_j\beta_j + \epsilon_j = Z_j\delta_j + \epsilon_j \tag{23.20}$$

where $\Gamma_j' = (-1, \gamma_j', 0)$, $B_j' = (\beta_j', 0)$, $Z_j' = (Y_j', X_j')$ and $\delta_j' = (\gamma_j', \beta_j')$. Y is the matrix of endogenous variables and X is the matrix of exogenous variables.

In the first stage, we regress the right-hand side endogenous variables y_j on all exogenous variables X and get the fitted values:

$$\hat{Y}_j = X(X'X)^{-1}X'Xy_j. \tag{23.21}$$

In the second stage, we regress y_j on $\hat{Y}_j$ and X_j to get:

$$\hat{\delta}_{2SLS} = (\hat{Z}_j'\hat{Z}_j)^{-1}\hat{Z}_j'y. \tag{23.22}$$

where $\hat{Z}_j = (\hat{Y}_j, X_j)$.

Weighted TSLS applies the weights in the second stage so that:

$$\hat{\delta}_{W2SLS} = (\hat{Z}_j'\hat{V}^{-1}\hat{Z}_j)^{-1}\hat{Z}_j'\hat{V}^{-1}y \tag{23.23}$$

where the elements of the variance matrix are estimated in the usual fashion using the residuals from unweighted TSLS.

If you choose to iterate the weights, X is estimated at each step using the current values of the coefficients and residuals.

Three-Stage Least Squares (3SLS)

Since TSLS is a single equation estimator that does not take account of the covariances between residuals, it is not, in general, fully efficient. 3SLS is a system method that estimates all of the coefficients of the model, then forms weights and reestimates the model using the estimated weighting matrix. It should be viewed as the endogenous variable analogue to the SUR estimator described above.

The first two stages of 3SLS are the same as in TSLS. In the third stage, we apply feasible generalized least squares (FGLS) to the equations in the system in a manner analogous to the SUR estimator.

SUR uses the OLS residuals to obtain a consistent estimate of the cross-equation covariance matrix Σ. This covariance estimator is not, however, consistent if any of the right-hand side variables are endogenous. 3SLS uses the 2SLS residuals to obtain a consistent estimate of Σ.

In the balanced case, we may write the equation as,

$$\hat{\delta}_{3SLS} = (Z(\hat{\Sigma}^{-1} \otimes X(X'X)^{-1}X')Z)^{-1}Z(\hat{\Sigma}^{-1} \otimes X(X'X)^{-1}X')y\,, \tag{23.24}$$

where $\hat{\Sigma}$ has typical element:

$$s_{ij} = ((y_i - Z_i\hat{\gamma}_{2SLS})'(y_j - Z_j\hat{\gamma}_{2SLS}))/\max(T_i, T_j)\,. \tag{23.25}$$

If you choose to iterate the weights, the current coefficients and residuals will be used to estimate $\hat{\Sigma}$.

Generalized Method of Moments (GMM)

The basic idea underlying GMM is simple and intuitive. We have a set of theoretical moment conditions that the parameters of interest θ should satisfy. We denote these moment conditions as:

$$\mathrm{E}(m(y, \theta)) = 0\,. \tag{23.26}$$

The method of moments estimator is defined by replacing the moment condition (23.26) by its sample analog:

$$(\sum_t m(y_t, \theta))/T = 0\,. \tag{23.27}$$

However, condition (23.27) will not be satisfied for any θ when there are more restrictions m than there are parameters θ. To allow for such overidentification, the GMM estimator is defined by minimizing the following criterion function:

$$\sum_t m(y_t, \theta) A(y_t, \theta) m(y_t, \theta) \tag{23.28}$$

which measures the "distance" between m and zero. A is a weighting matrix that weights each moment condition. Any symmetric positive definite matrix A will yield a consistent estimate of θ. However, it can be shown that a necessary (but not sufficient) condition to obtain an (asymptotically) efficient estimate of θ is to set A equal to the inverse of the covariance matrix Ω of the sample moments m. This follows intuitively, since we want to put less weight on the conditions that are more imprecise.

To obtain GMM estimates in EViews, you must be able to write the moment conditions in Equation (23.26) as an orthogonality condition between the residuals of a regression equation, $u(y, \theta, X)$, and a set of instrumental variables, Z, so that:

$$m(\theta, y, X, Z) = Z'u(\theta, y, X) \tag{23.29}$$

For example, the OLS estimator is obtained as a GMM estimator with the orthogonality conditions:

$$X'(y - X\beta) = 0\,. \tag{23.30}$$

For the GMM estimator to be identified, there must be at least as many instrumental variables Z as there are parameters θ. See the section on "Generalized Method of Moments (GMM)" beginning on page 488 for additional examples of GMM orthogonality conditions.

An important aspect of specifying a GMM problem is the choice of the weighting matrix A. EViews uses the optimal $A = \hat{\Omega}^{-1}$, where $\hat{\Omega}$ is the estimated covariance matrix of the sample moments m. EViews uses the consistent TSLS estimates for the initial estimate of θ in forming the estimate of Ω.

White's Heteroskedasticity Consistent Covariance Matrix

If you choose the **GMM-Cross section** option, EViews estimates Ω using White's heteroskedasticity consistent covariance matrix:

$$\hat{\Omega}_W = \hat{\Gamma}(0) = \frac{1}{T-k}\left(\sum_{t=1}^{T} Z_t' u_t u_t' Z_t\right) \tag{23.31}$$

where u is the vector of residuals, and Z_t is a $k \times p$ matrix such that the p moment conditions at t may be written as $m(\theta, y_t, X_t, Z_t) = Z_t' u(\theta, y_t, X_t)$.

Heteroskedasticity and Autocorrelation Consistent (HAC) Covariance Matrix

If you choose the **GMM-Time series** option, EViews estimates Ω by,

$$\hat{\Omega}_{HAC} = \hat{\Gamma}(0) + \left(\sum_{j=1}^{T-1} k(j,q)(\hat{\Gamma}(j) + \hat{\Gamma}'(j)) \right) \tag{23.32}$$

where:

$$\hat{\Gamma}(j) = \frac{1}{T-k} \left(\sum_{t=j+1}^{T} Z_{t-j}' u_t u_{t-j}' Z_t \right). \tag{23.33}$$

You also need to specify the *kernel* κ and the *bandwidth* q.

Kernel Options

The kernel κ is used to weight the covariances so that $\hat{\Omega}$ is ensured to be positive semi-definite. EViews provides two choices for the kernel, Bartlett and quadratic spectral (QS). The Bartlett kernel is given by,

$$\kappa(j,q) = \begin{cases} 1-(j/q) & 0 \le j \le q \\ 0 & \text{otherwise} \end{cases} \tag{23.34}$$

while the quadratic spectral (QS) kernel is given by:

$$k(j/q) = \frac{25}{12(\pi x)^2} \left(\frac{\sin(6\pi x/5)}{6\pi x/5} - \cos(6\pi x/5) \right) \tag{23.35}$$

where $x = j/q$. The QS has a faster rate of convergence than the Bartlett and is smooth and not truncated (Andrews 1991). Note that even though the QS kernel is not truncated, it still depends on the bandwidth q (which need not be an integer).

Bandwidth Selection

The bandwidth q determines how the weights given by the kernel change with the lags in the estimation of Ω. Newey-West fixed bandwidth is based solely on the number of observations in the sample and is given by:

$$q = \text{int}(4(T/100)^{2/9}) \tag{23.36}$$

where int( ) denotes the integer part of the argument.

EViews also provides two "automatic", or data dependent bandwidth selection methods that are based on the autocorrelations in the data. Both methods select the bandwidth according to:

$$q = \begin{cases} \text{int}(1.1447(\hat{\alpha}(1)T)^{1/3}) & \text{for the Bartlett kernel} \\ 1.3221(\hat{\alpha}(2)T)^{1/5} & \text{for the QS kernel} \end{cases} \tag{23.37}$$

The two methods, Andrews and Variable-Newey-West, differ in how they estimate $\hat{\alpha}(1)$ and $\hat{\alpha}(2)$.

Andrews (1991) is a parametric method that assumes the sample moments follow an AR(1) process. We first fit an AR(1) to each sample moment (23.29) and estimate the autocorrelation coefficients $\hat{\rho}_i$ and the residual variances $\hat{\sigma}_i^2$ for $i = 1, 2, \ldots, zn$, where z is the number of instrumental variables and n is the number of equations in the system. Then $\hat{\alpha}(1)$ and $\hat{\alpha}(2)$ are estimated by:

$$\begin{aligned} \hat{\alpha}(1) &= \left(\sum_{i=1}^{zn} \frac{4\hat{\rho}_i^2\hat{\sigma}_i^4}{(1-\hat{\rho}_i)^6(1+\hat{\rho}_i)^2} \right) \Big/ \left(\sum_{i=1}^{zn} \frac{\hat{\sigma}_i^4}{(1-\hat{\rho}_i)^4} \right) \\ \hat{\alpha}(2) &= \left(\sum_{i=1}^{zn} \frac{4\hat{\rho}_i^2\hat{\sigma}_i^4}{(1-\hat{\rho}_i)^8} \right) \Big/ \left(\sum_{i=1}^{zn} \frac{\hat{\sigma}_i^4}{(1-\hat{\rho}_i)^4} \right) \end{aligned} \tag{23.38}$$

Note that we weight all moments equally, including the moment corresponding to the constant.

Newey-West (1994) is a nonparametric method based on a truncated weighted sum of the estimated cross-moments $\hat{\Gamma}(j)$. $\hat{\alpha}(1)$ and $\hat{\alpha}(2)$ are estimated by,

$$\hat{\alpha}(p) = \left(\frac{l'F(p)l}{l'F(0)l} \right) \tag{23.39}$$

where l is a vector of ones and:

$$F(p) = \Gamma(0) + \sum_{i=1}^{L} i^p(\hat{\Gamma}(i) + \hat{\Gamma}'(i)), \tag{23.40}$$

for $p = 1, 2$.

One practical problem with the Newey-West method is that we have to choose a lag selection parameter L. The choice of L is arbitrary, subject to the condition that it grow at a certain rate. EViews sets the lag parameter to

$$L = \begin{cases} \text{int}(4(T/100)^{2/9}) & \text{for the Bartlett kernel} \\ T & \text{for the QS kernel} \end{cases} \tag{23.41}$$

Prewhitening

You can also choose to prewhiten the sample moments m to "soak up" the correlations in m prior to GMM estimation. We first fit a VAR(1) to the sample moments:

$$m_t = Am_{t-1} + v_t. \tag{23.42}$$

Then the variance $\hat{\Omega}$ of m is estimated by $\hat{\Omega} = (I - A)^{-1}\hat{\Omega}^*(I - A)^{-1}$ where $\hat{\Omega}^*$ is the variance of the residuals v_t and is computed using any of the above methods. The GMM estimator is then found by minimizing the criterion function:

$$u'Z\hat{\Omega}^{-1}Z'u \tag{23.43}$$

Note that while Andrews and Monahan (1992) adjust the VAR estimates to avoid singularity when the moments are near unit root processes, EViews does not perform this eigenvalue adjustment.

Chapter 24. Vector Autoregression and Error Correction Models

The structural approach to time series modeling uses economic theory to model the relationship among the variables of interest. Unfortunately, economic theory is often not rich enough to provide a dynamic specification that identifies all of these relationships. Furthermore, estimation and inference are complicated by the fact that endogenous variables may appear on both the left and right sides of equations.

These problems lead to alternative, non-structural approaches to modeling the relationship among several variables. This chapter describes the estimation and analysis of vector autoregression (VAR) and the vector error correction (VEC) models. We also describe tools for testing the presence of cointegrating relationships among several non-stationary variables.

Vector Autoregressions (VARs)

The vector autoregression (VAR) is commonly used for forecasting systems of interrelated time series and for analyzing the dynamic impact of random disturbances on the system of variables. The VAR approach sidesteps the need for structural modeling by treating every endogenous variable in the system as a function of the lagged values of all of the endogenous variables in the system.

The mathematical representation of a VAR is:

$$y_t = A_1 y_{t-1} + \cdots + A_p y_{t-p} + B x_t + \epsilon_t \tag{24.1}$$

where y_t is a k vector of endogenous variables, x_t is a d vector of exogenous variables, A_1, ..., A_p and B are matrices of coefficients to be estimated, and ϵ_t is a vector of innovations that may be contemporaneously correlated but are uncorrelated with their own lagged values and uncorrelated with all of the right-hand side variables.

Since only lagged values of the endogenous variables appear on the right-hand side of the equations, simultaneity is not an issue and OLS yields consistent estimates. Moreover, even though the innovations ϵ_t may be contemporaneously correlated, OLS is efficient and equivalent to GLS since all equations have identical regressors.

As an example, suppose that industrial production (IP) and money supply (M1) are jointly determined by a VAR and let a constant be the only exogenous variable. Assuming that the VAR contains two lagged values of the endogenous variables, it may be written as:

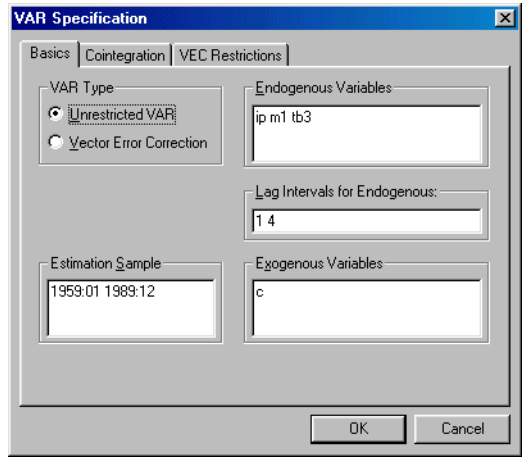
$$\begin{aligned} IP_t &= a_{11} IP_{t-1} + a_{12} M1_{t-1} + b_{11} IP_{t-2} + b_{12} M1_{t-2} + c_1 + \epsilon_{1t} \\ M1_t &= a_{21} IP_{t-1} + a_{22} M1_{t-1} + b_{21} IP_{t-2} + b_{22} M1_{t-2} + c_2 + \epsilon_{2t} \end{aligned} \tag{24.2}$$

where a_{ij}, b_{ij}, c_i are the parameters to be estimated.

Estimating a VAR in EViews

To specify a VAR in EViews, you must first create a var object. Select **Quick/Estimate VAR...** or type `var` in the command window. The **Basics** tab of the VAR Specification dialog will prompt you to define the structure of your VAR.

You should fill out the dialog with the appropriate information:



- Select the VAR type: **Unrestricted VAR** or **Vector Error Correction** (VEC). What we have been calling a VAR is actually an *unrestricted* VAR. VECs are explained below.
- Set the estimation sample.
- Enter the lag specification in the appropriate edit box. This information is entered in pairs: each pair of numbers defines a *range* of lags. For example, the lag pair shown above:

  ```
  1 4
  ```

  tells EViews to use the first *through* fourth lags of all the endogenous variables in the system as right-hand side variables.

  You can add any number of lag intervals, all entered in pairs. The lag specification:

  ```
  2 4 6 9 12 12
  ```

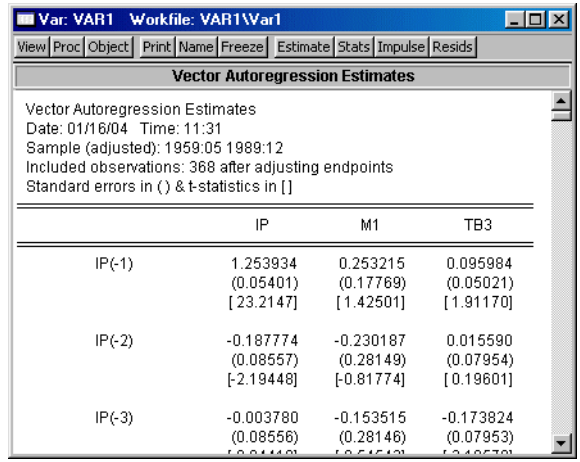
  uses lags 2–4, 6–9, and 12.

- Enter the names of endogenous and exogenous series in the appropriate edit boxes. Here we have listed M1, IP, and TB3 as endogenous series, and have used the special series `C` as the constant exogenous term. If either list of series were longer, we could have created a named group object containing the list and then entered the group name.

The remaining dialog tabs (**Cointegration** and **Restrictions**) are relevant only for VEC models and are explained below.

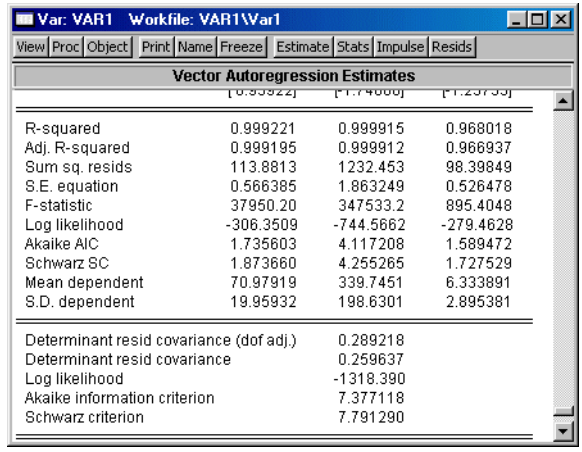
VAR Estimation Output

Once you have specified the VAR, click **OK**. EViews will display the estimation results in the VAR window.


Var: VAR1 Workfile: VAR1\Var1
View Proc Object Print Name Freeze Estimate Stats Impulse Resids
Vector Autoregression Estimates
Vector Autoregression Estimates
Date: 01/16/04 Time: 11:31
Sample (adjusted): 1959:05 1989:12
Included observations: 368 after adjusting endpoints
Standard errors in ( ) & t-statistics in [ ]
IP M1 TB3
IP(-1) 1.253934 0.253215 0.095984
(0.05401) (0.17769) (0.05021)
[23.2147] [1.42501] [1.91170]
IP(-2) -0.187774 -0.230187 0.015590
(0.08557) (0.28149) (0.07954)
[-2.19448] [-0.81774] [0.19601]
IP(-3) -0.003780 -0.153515 -0.173824
(0.08556) (0.28146) (0.07953)

Each column in the table corresponds to an equation in the VAR. For each right-hand side variable, EViews reports the estimated coefficient, its standard error, and the t-statistic. For example, the coefficient for IP(-1) in the TB3 equation is 0.095984.

EViews displays additional information below the coefficient summary. The first part of the additional output presents standard OLS regression statistics for each equation. The results are computed separately for each equation using the appropriate residuals and are displayed in the corresponding column. The numbers at the very bottom of the table are the summary statistics for the VAR system as a whole.


Var: VAR1 Workfile: VAR1\Var1
View Proc Object Print Name Freeze Estimate Stats Impulse Resids
Vector Autoregression Estimates
R-squared 0.999221 0.999915 0.968018
Adj. R-squared 0.999195 0.999912 0.966937
Sum sq. resids 113.8813 1232.453 98.39849
S.E. equation 0.566385 1.863249 0.526478
F-statistic 37950.20 347533.2 895.4048
Log likelihood -306.3509 -744.5662 -279.4628
Akaike AIC 1.735603 4.117208 1.589472
Schwarz SC 1.873660 4.255265 1.727529
Mean dependent 70.97919 339.7451 6.333891
S.D. dependent 19.95932 198.6301 2.895381
Determinant resid covariance (dof adj.) 0.289218
Determinant resid covariance 0.259637
Log likelihood -1318.390
Akaike information criterion 7.377118
Schwarz criterion 7.791290

The determinant of the residual covariance (degree of freedom adjusted) is computed as:

$$|\hat{\Omega}| = \det\left(\frac{1}{T-p}\sum_t \hat{\epsilon}_t \hat{\epsilon}_t'\right) \tag{24.3}$$

where p is the number of parameters per equation in the VAR. The unadjusted calculation ignores the p. The log likelihood value is computed assuming a multivariate normal (Gaussian) distribution as:

$$l = -\frac{T}{2}\{k(1+\log 2\pi) + \log\left|\hat{\Omega}\right|\} \tag{24.4}$$

The two information criteria are computed as:

$$\begin{aligned} AIC &= -2l/T + 2n/T \\ SC &= -2l/T + n\log T/T \end{aligned} \tag{24.5}$$

where $n = k(d+pk)$ is the total number of estimated parameters in the VAR. These information criteria can be used for model selection such as determining the lag length of the VAR, with smaller values of the information criterion being preferred. It is worth noting that some reference sources may define the AIC/SC differently, either omitting the "inessential" constant terms from the likelihood, or not dividing by T (see also Appendix E, "Information Criteria", on page 971 for additional discussion of information criteria).

Views and Procs of a VAR

Once you have estimated a VAR, EViews provides various views to work with the estimated VAR. In this section, we discuss views that are specific to VARs. For other views and procedures, see the general discussion of system views in Chapter 23, "System Estimation", beginning on page 695.

Diagnostic Views

A set of diagnostic views are provided under the menus **View/Lag Structure** and **View/Residual Tests** in the VAR window. These views should help you check the appropriateness of the estimated VAR.

Lag Structure

AR Roots Table/Graph

Reports the *inverse* roots of the characteristic AR polynomial; see Lütkepohl (1991). The estimated VAR is stable (stationary) if all roots have modulus less than one and lie inside the unit circle. If the VAR is not stable, certain results (such as impulse response standard errors) are not valid. There will be kp roots, where k is the number of endogenous variables and p is the largest lag. If you estimated a VEC with r cointegrating relations, $k-r$ roots should be equal to unity.

Pairwise Granger Causality Tests

Carries out pairwise Granger causality tests and tests whether an endogenous variable can be treated as exogenous. For each equation in the VAR, the output displays χ^2 (Wald) sta-

tistics for the joint significance of each of the other lagged endogenous variables in that equation. The statistic in the last row (**All**) is the χ^2 statistic for joint significance of all other lagged endogenous variables in the equation.

Warning: if you have estimated a VEC, the lagged variables that are tested for exclusion are only those that are first differenced. The lagged level terms in the cointegrating equations (the error correction terms) are not tested.

Lag Exclusion Tests

Carries out lag exclusion tests for each lag in the VAR. For each lag, the χ^2 (Wald) statistic for the joint significance of all endogenous variables at that lag is reported for each equation separately and jointly (last column).

Lag Length Criteria

Computes various criteria to select the lag order of an unrestricted VAR. You will be prompted to specify the maximum lag to "test" for. The table displays various information criteria for all lags up to the specified maximum. (If there are no exogenous variables in the VAR, the lag starts at 1; otherwise the lag starts at 0.) The table indicates the selected lag from each column criterion by an asterisk "*". For columns 4–7, these are the lags with the smallest value of the criterion.

All the criteria are discussed in Lütkepohl (1991, Section 4.3). The sequential modified likelihood ratio (LR) test is carried out as follows. Starting from the maximum lag, test the hypothesis that the coefficients on lag l are jointly zero using the χ^2 statistics:

$$LR = (T-m)\{\log|\Omega_{\ell-1}| - \log|\Omega_{\ell}|\} \sim \chi^2(k^2) \tag{24.6}$$

where m is the number of parameters per equation under the alternative. Note that we employ Sims' (1980) small sample modification which uses $(T-m)$ rather than T. We compare the modified LR statistics to the 5% critical values starting from the maximum lag, and decreasing the lag one at a time until we first get a rejection. The alternative lag order from the first rejected test is marked with an asterisk (if no test rejects, the minimum lag will be marked with an asterisk). It is worth emphasizing that even though the individual tests have size 0.05, the overall size of the test will not be 5%; see the discussion in Lütkepohl (1991, pp. 125–126).

Residual Tests

Correlograms

Displays the pairwise cross-correlograms (sample autocorrelations) for the estimated residuals in the VAR for the specified number of lags. The cross-correlograms can be displayed in three different formats. There are two tabular forms, one ordered by variables (**Tabulate by Variable**) and one ordered by lags (**Tabulate by Lag**). The **Graph** form displays a matrix of pairwise cross-correlograms. The dotted line in the graphs represent plus or

minus two times the asymptotic standard errors of the lagged correlations (computed as $1/\sqrt{T}$.

Portmanteau Autocorrelation Test

Computes the multivariate Box-Pierce/Ljung-Box Q-statistics for residual serial correlation up to the specified order (see Lütkepohl, 1991, 4.4.21 & 4.4.23 for details). We report both the Q-statistics and the adjusted Q-statistics (with a small sample correction). Under the null hypothesis of no serial correlation up to lag h, both statistics are *approximately* distributed χ^2 with degrees of freedom $k^2(h-p)$ where p is the VAR lag order. The asymptotic distribution is approximate in the sense that it requires the MA coefficients to be zero for lags $i > h-p$. Therefore, this approximation will be poor if the roots of the AR polynomial are close to one and h is small. In fact, the degrees of freedom becomes negative for $h < p$.

Autocorrelation LM Test

Reports the multivariate LM test statistics for residual serial correlation up to the specified order. The test statistic for lag order h is computed by running an auxiliary regression of the residuals u_t on the original right-hand regressors and the lagged residual u_{t-h}, where the missing first h values of u_{t-h} are filled with zeros. See Johansen (1995a, p. 22) for the formula of the LM statistic. Under the null hypothesis of no serial correlation of order h, the LM statistic is asymptotically distributed χ^2 with k^2 degrees of freedom.

Normality Test

Reports the multivariate extensions of the Jarque-Bera residual normality test, which compares the third and fourth moments of the residuals to those from the normal distribution. For the multivariate test, you must choose a factorization of the k residuals that are orthogonal to each other (see "Impulse Responses" on page 729 for additional discussion of the need for orthogonalization).

Let P be a $k \times k$ factorization matrix such that:

$$v_t = Pu_t \sim N(0, I_k) \tag{24.7}$$

where u_t is the demeaned residuals. Define the third and fourth moment vectors $m_3 = \sum_t v_t^3/T$ and $m_4 = \sum_t v_t^4/T$. Then:

$$\sqrt{T}\begin{bmatrix} m_3 \\ m_4 - 3 \end{bmatrix} \rightarrow N\left(0, \begin{bmatrix} 6I_k & 0 \\ 0 & 24I_k \end{bmatrix}\right) \tag{24.8}$$

under the null hypothesis of normal distribution. Since each component is independent of each other, we can form a χ^2 statistic by summing squares of any of these third and fourth moments.

EViews provides you with choices for the factorization matrix P:

- **Cholesky** (Lütkepohl 1991, pp. 155-158): P is the inverse of the lower triangular Cholesky factor of the residual covariance matrix. The resulting test statistics depend on the ordering of the variables in the VAR.

- **Inverse Square Root of Residual Correlation Matrix** (Doornik and Hansen 1994): $P = H\Lambda^{-1/2}H'V$ where Λ is a diagonal matrix containing the eigenvalues of the residual correlation matrix on the diagonal, H is a matrix whose columns are the corresponding eigenvectors, and V is a diagonal matrix containing the inverse square root of the residual variances on the diagonal. This P is essentially the inverse square root of the residual correlation matrix. The test is invariant to the ordering and to the scale of the variables in the VAR. As suggested by Doornik and Hansen (1994), we perform a small sample correction to the transformed residuals v_t before computing the statistics.

- **Inverse Square Root of Residual Covariance Matrix** (Urzua 1997): $P = GD^{-1/2}G'$ where D is the diagonal matrix containing the eigenvalues of the residual covariance matrix on the diagonal and G is a matrix whose columns are the corresponding eigenvectors. This test has a specific alternative, which is the quartic exponential distribution. According to Urzua, this is the "most likely" alternative to the multivariate normal with finite fourth moments since it can approximate the multivariate Pearson family "as close as needed." As recommended by Urzua, we make a small sample correction to the transformed residuals v_t before computing the statistics. This small sample correction differs from the one used by Doornik and Hansen (1994); see Urzua (1997, Section D).

- **Factorization from Identified (Structural) VAR**: $P = B^{-1}A$ where A, B are estimated from the structural VAR model. This option is available only if you have estimated the factorization matrices A and B using the structural VAR (see page 733, below).

EViews reports test statistics for each orthogonal component (labeled RESID1, RESID2, and so on) and for the joint test. For individual components, the estimated skewness m_3 and kurtosis m_4 are reported in the first two columns together with the p-values from the $\chi^2(1)$ distribution (in square brackets). The Jarque-Bera column reports:

$$T\left\{\frac{m_3^2}{6}+\frac{(m_4-3)^2}{24}\right\} \tag{24.9}$$

with p-values from the $\chi^2(2)$ distribution. *Note: in contrast to the Jarque-Bera statistic computed in the series view, this statistic is not computed using a degrees of freedom correction.*

For the joint tests, we will generally report:

$$\begin{aligned}\lambda_3 &= Tm_3'm_3/6 \rightarrow \chi^2(k)\\ \lambda_4 &= T(m_4-3)'(m_4-3)/24 \rightarrow \chi^2(k)\\ \lambda &= \lambda_3+\lambda_4 \rightarrow \chi^2(2k).\end{aligned} \tag{24.10}$$

If, however, you choose Urzua's (1997) test, λ will not only use the sum of squares of the "pure" third and fourth moments but will also include the sum of squares of all *cross* third and fourth moments. In this case, λ is asymptotically distributed as a χ^2 with $k(k+1)(k+2)(k+7)/24$ degrees of freedom.

White Heteroskedasticity Test

These tests are the extension of White's (1980) test to systems of equations as discussed by Kelejian (1982) and Doornik (1995). The test regression is run by regressing each cross product of the residuals on the cross products of the regressors and testing the joint significance of the regression. The **No Cross Terms** option uses only the levels and squares of the original regressors, while the **With Cross Terms** option includes all non-redundant cross-products of the original regressors in the test equation. The test regression always includes a constant term as a regressor.

The first part of the output displays the joint significance of the regressors excluding the constant term for each test regression. You may think of each test regression as testing the constancy of each element in the residual covariance matrix separately. Under the null of no heteroskedasticity or (no misspecification), the non-constant regressors should not be jointly significant.

The last line of the output table shows the LM chi-square statistics for the joint significance of all regressors in the system of test equations (see Doornik, 1995, for details). The system LM statistic is distributed as a χ^2 with degrees of freedom mn, where $m = k(k+1)/2$ is the number of cross-products of the residuals in the system and n is the number of the common set of right-hand side variables in the test regression.

Notes on Comparability

Many of the diagnostic tests given above may be computed "manually" by estimating the VAR using a system object and selecting **View/Wald Coefficient Tests...** We caution you that the results from the system will not match those from the VAR diagnostic views for various reasons:

- The system object will, in general, use the maximum possible observations for each equation in the system. By contrast, VAR objects force a balanced sample in case there are missing values.
- The estimates of the weighting matrix used in system estimation do not contain a degrees of freedom correction (the residual sums-of-squares are divided by T rather

than by $T-k$), while the VAR estimates do perform this adjustment. Even though estimated using comparable specifications and yielding identifiable coefficients, the test statistics from system SUR and the VARs will show small (asymptotically insignificant) differences.

Impulse Responses

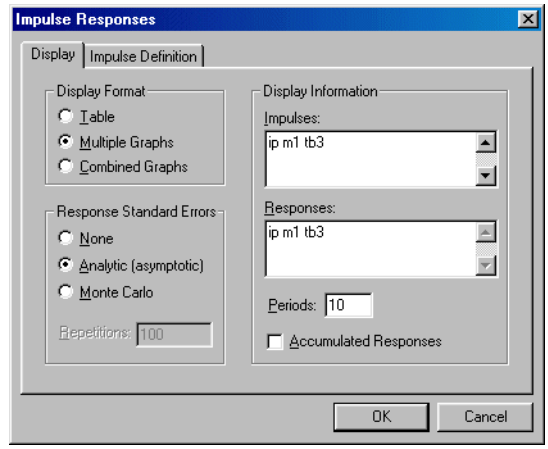
A shock to the i-th variable not only directly affects the i-th variable but is also transmitted to all of the other endogenous variables through the dynamic (lag) structure of the VAR. An impulse response function traces the effect of a one-time shock to one of the innovations on current and future values of the endogenous variables.

If the innovations ϵ_t are contemporaneously uncorrelated, interpretation of the impulse response is straightforward. The i-th innovation $\epsilon_{i,t}$ is simply a shock to the i-th endogenous variable $y_{i,t}$. Innovations, however, are usually correlated, and may be viewed as having a common component which cannot be associated with a specific variable. In order to interpret the impulses, it is common to apply a transformation P to the innovations so that they become uncorrelated:

$$v_t = P\epsilon_t \sim (0, D) \tag{24.11}$$

where D is a *diagonal* covariance matrix. As explained below, EViews provides several options for the choice of P.

To obtain the impulse response functions, first estimate a VAR. Then select **View/Impulse Response...** from the VAR toolbar. You will see a dialog box with two tabs: **Display** and **Impulse Definition**.



The **Display** tab provides the following options:

- **Display Format**: displays results as a table or graph. Keep in mind that if you choose the **Combined Graphs** option, the **Response Standard Errors** option will be ignored and the standard errors will not be displayed. Note also that the output table format is ordered by response variables, not by impulse variables.

- **Display Information**: you should enter the variables for which you wish to generate innovations (**Impulses**) and the variables for which you wish to observe the

responses (**Responses**). You may either enter the name of the endogenous variables or the numbers corresponding to the ordering of the variables. For example, if you specified the VAR as GDP, M1, CPI, then you may either type,

```
GDP CPI M1
```

or,

```
1 3 2
```

The order in which you enter these variables only affects the display of results.

You should also specify a positive integer for the number of periods to trace the response function. To display the accumulated responses, check the **Accumulate Response** box. For stationary VARs, the impulse responses should die out to zero and the accumulated responses should asymptote to some (non-zero) constant.

- **Response Standard Errors**: provides options for computing the response standard errors. Note that analytic and/or Monte Carlo standard errors are currently not available for certain **Impulse** options and for vector error correction (VEC) models. If you choose **Monte Carlo** standard errors, you should also specify the number of repetitions to use in the appropriate edit box.

  If you choose the table format, the estimated standard errors will be reported in parentheses below the responses. If you choose to display the results in multiple graphs, the graph will contain the plus/minus two standard error bands about the impulse responses. The standard error bands are not displayed in combined graphs.

The **Impulse** tab provides the following options for transforming the impulses:

- **Residual—One Unit** sets the impulses to one unit of the residuals. This option ignores the units of measurement and the correlations in the VAR residuals so that no transformation is performed. The responses from this option are the MA coefficients of the infinite MA order Wold representation of the VAR.
- **Residual—One Std. Dev.** sets the impulses to one standard deviation of the residuals. This option ignores the correlations in the VAR residuals.
- **Cholesky** uses the inverse of the Cholesky factor of the residual covariance matrix to orthogonalize the impulses. This option imposes an ordering of the variables in the VAR and attributes all of the effect of any common component to the variable that comes first in the VAR system. Note that responses can change dramatically if you change the ordering of the variables. You may specify a different VAR ordering by reordering the variables in the **Cholesky Ordering** edit box.

  The **(d.f. adjustment)** option makes a small sample degrees of freedom correction when estimating the residual covariance matrix used to derive the Cholesky factor. The (i,j)-th element of the residual covariance matrix with degrees of freedom cor-

rection is computed as $\sum_t e_{i,t} e_{j,t}/(T-p)$ where p is the number of parameters per equation in the VAR. The **(no d.f. adjustment)** option estimates the (i,j)-th element of the residual covariance matrix as $\sum_t e_{i,t} e_{j,t}/T$. *Note: early versions of EViews computed the impulses using the Cholesky factor from the residual covariance matrix with no degrees of freedom adjustment.*

- **Generalized Impulses** as described by Pesaran and Shin (1998) constructs an orthogonal set of innovations that does not depend on the VAR ordering. The generalized impulse responses from an innovation to the j-th variable are derived by applying a variable specific Cholesky factor computed with the j-th variable at the top of the Cholesky ordering.

- **Structural Decomposition** uses the orthogonal transformation estimated from the structural factorization matrices. This approach is not available unless you have estimated the structural factorization matrices as explained in "Structural (Identified) VARs" on page 733.

- **User Specified** allows you to specify your own impulses. Create a matrix (or vector) that contains the impulses and type the name of that matrix in the edit box. If the VAR has k endogenous variables, the impulse matrix must have k rows and 1 or k columns, where each column is a impulse vector.

  For example, say you have a $k = 3$ variable VAR and wish to apply simultaneously a positive one unit shock to the first variable and a negative one unit shock to the second variable. Then you will create a 3×1 impulse matrix containing the values 1, -1, and 0. Using commands, you can enter:

  ```
  matrix(3,1) shock
  shock.fill(by=c) 1,-1,0
  ```

  and type the name of the matrix SHOCK in the edit box.

Variance Decomposition

While impulse response functions trace the effects of a shock to one endogenous variable on to the other variables in the VAR, *variance decomposition* separates the variation in an endogenous variable into the component shocks to the VAR. Thus, the variance decomposition provides information about the relative importance of each random innovation in affecting the variables in the VAR.

To obtain the variance decomposition, select **View/Variance Decomposition...** from the var object toolbar. You should provide the same information as for impulse responses above. Note that since non-orthogonal factorization will yield decompositions that do not satisfy an adding up property, your choice of factorization is limited to orthogonal factorizations.

The table format displays a separate variance decomposition for each endogenous variable. The second column, labeled "S.E.", contains the forecast error of the variable at the given forecast horizon. The source of this forecast error is the variation in the current and future values of the innovations to each endogenous variable in the VAR. The remaining columns give the percentage of the forecast variance due to each innovation, with each row adding up to 100.

As with the impulse responses, the variance decomposition based on the Cholesky factor can change dramatically if you alter the ordering of the variables in the VAR. For example, the first period decomposition for the first variable in the VAR ordering is completely due to its own innovation.

Factorization based on structural orthogonalization is available only if you have estimated the structural factorization matrices as explained in "Structural (Identified) VARs" on page 733. Note that the forecast standard errors should be identical to those from the Cholesky factorization *if the structural VAR is just identified*. For over-identified structural VARs, the forecast standard errors may differ in order to maintain the adding up property.

Procs of a VAR

Most of the procedures available for a VAR are common to those available for a system object (see "System Procs" on page 707). Here, we discuss only those procedures that are unique to the VAR object.

Make System

This proc creates a system object that contains an equivalent VAR specification. If you want to estimate a non-standard VAR, you may use this proc as a quick way to specify a VAR in a system object which you can then modify to meet your needs. For example, while the VAR object requires each equation to have the same lag structure, you may want to relax this restriction. To estimate a VAR with unbalanced lag structure, use the **Proc/Make System** procedure to create a VAR system with a balanced lag structure and edit the system specification to meet the desired lag specification.

The **By Variable** option creates a system whose specification (and coefficient number) is ordered by variables. Use this option if you want to edit the specification to exclude lags of a specific variable from some of the equations. The **By Lag** option creates a system whose specification (and coefficient number) is ordered by lags. Use this option if you want to edit the specification to exclude certain lags from some of the equations.

For vector error correction (VEC) models, treating the coefficients of the cointegrating vector as additional unknown coefficients will make the resulting system unidentified. In this case, EViews will create a system object where the coefficients for the cointegrating vectors are fixed at the estimated values from the VEC. If you want to estimate the coefficients of

the cointegrating vector in the system, you may edit the specification, but you should make certain that the resulting system is identified.

You should also note that while the standard VAR can be estimated efficiently by equation-by-equation OLS, this is generally not the case for the modified specification. You may wish to use one of the system-wide estimation methods (e.g. SUR) when estimating non-standard VARs using the system object.

Estimate Structural Factorization

This procedure is used to estimate the factorization matrices for a structural (or identified) VAR. The full details of this procedure is given in "Structural (Identified) VARs" on page 733. You must first estimate the structural factorization matrices using this proc in order to use the structural options in impulse responses and variance decompositions.

Structural (Identified) VARs

The main purpose of structural VAR (SVAR) estimation is to obtain non-recursive orthogonalization of the error terms for impulse response analysis. This alternative to the recursive Cholesky orthogonalization requires the user to impose enough restrictions to identify the orthogonal (structural) components of the error terms.

Let y_t be a k-element vector of the endogenous variables and let $\Sigma = E[e_t e_t']$ be the residual covariance matrix. Following Amisano and Giannini (1997), the class of SVAR models that EViews estimates may be written as:

$$Ae_t = Bu_t \tag{24.12}$$

where e_t and u_t are vectors of length k. e_t is the observed (or reduced form) residuals, while u_t is the unobserved structural innovations. A and B are $k \times k$ matrices to be estimated. The structural innovations u_t are assumed to be orthonormal, *i.e.* its covariance matrix is an identity matrix $E[u_t u_t'] = I$. The assumption of orthonormal innovations u_t imposes the following identifying restrictions on A and B:

$$A\Sigma A' = BB'. \tag{24.13}$$

Noting that the expressions on either side of (24.13) are symmetric, this imposes $k(k+1)/2$ restrictions on the $2k^2$ unknown elements in A and B. Therefore, in order to identify A and B, you need to supply at least $2k^2 - k(k+1)/2 = k(3k-1)/2$ additional restrictions.

Specifying the Identifying Restrictions

As explained above, in order to estimate the orthogonal factorization matrices A and B, you need to provide additional identifying restrictions. We distinguish two types of identi-

fying restrictions: *short-run* and *long-run*. For either type, the identifying restrictions can be specified either in text form or by pattern matrices.

Short-run Restrictions by Pattern Matrices

For many problems, the identifying restrictions on the A and B matrices are simple zero exclusion restrictions. In this case, you can specify the restrictions by creating a named "pattern" matrix for A and B. Any elements of the matrix that you want to be estimated should be assigned a missing value "NA". All non-missing values in the pattern matrix will be held fixed at the specified values.

For example, suppose you want to restrict A to be a lower triangular matrix with ones on the main diagonal and B to be a diagonal matrix. Then the pattern matrices (for a $k = 3$ variable VAR) would be:

$$A = \begin{pmatrix} 1 & 0 & 0 \\ \text{NA} & 1 & 0 \\ \text{NA} & \text{NA} & 1 \end{pmatrix}, \qquad B = \begin{pmatrix} \text{NA} & 0 & 0 \\ 0 & \text{NA} & 0 \\ 0 & 0 & \text{NA} \end{pmatrix}. \tag{24.14}$$

You can create these matrices interactively. Simply use **Object/New Object...** to create two new 3×3 matrices, A and B, and then use the spreadsheet view to edit the values. Alternatively, you can issue the following commands:

```
matrix(3,3) pata
' fill matrix in row major order
pata.fill(by=r) 1,0,0, na,1,0, na,na,1
matrix(3,3) patb = 0
patb(1,1) = na
patb(2,2) = na
patb(3,3) = na
```

Once you have created the pattern matrices, select **Proc/Estimate Structural Factorization...** from the VAR window menu. In the **SVAR Options** dialog, click the **Matrix** button and the **Short-Run Pattern** button and type in the name of the pattern matrices in the relevant edit boxes.

Short-run Restrictions in Text Form

For more general restrictions, you can specify the identifying restrictions in text form. In text form, you will write out the relation $Ae_t = Bu_t$ as a set of equations, identifying each element of the e_t and u_t vectors with special symbols. Elements of the A and B matrices to be estimated must be specified as elements of a coefficient vector.

To take an example, suppose again that you have a $k = 3$ variable VAR where you want to restrict A to be a lower triangular matrix with ones on the main diagonal and B to be a diagonal matrix. Under these restrictions, the relation $Ae_t = Bu_t$ can be written as:

$$\begin{aligned} e_1 &= b_{11}u_1 \\ e_2 &= -a_{21}e_1 + b_{22}u_2 \\ e_3 &= -a_{31}e_1 - a_{32}e_2 + b_{33}u_3 \end{aligned} \tag{24.15}$$

To specify these restrictions in text form, select **Proc/Estimate Structural Factorization...** from the VAR window and click the **Text** button. In the edit window, you should type the following:

```
@e1 = c(1)*@u1
@e2 = -c(2)*@e1 + c(3)*@u2
@e3 = -c(4)*@e1 - c(5)*@e2 + c(6)*@u3
```

The special key symbols "@e1", "@e2", "@e3," represent the first, second, and third elements of the e_t vector, while "@u1," "@u2", "@u3" represent the first, second, and third elements of the u_t vector. In this example, all unknown elements of the A and B matrices are represented by elements of the C coefficient vector.

Long-run Restrictions

The identifying restrictions embodied in the relation $Ae = Bu$ are commonly referred to as short-run restrictions. Blanchard and Quah (1989) proposed an alternative identification method based on restrictions on the long-run properties of the impulse responses. The (accumulated) long-run response C to structural innovations takes the form:

$$C = \hat{\Psi}_\infty A^{-1} B \tag{24.16}$$

where $\hat{\Psi}_\infty = (I - \hat{A}_1 - \cdots - \hat{A}_p)^{-1}$ is the estimated accumulated responses to the reduced form (observed) shocks. Long-run identifying restrictions are specified in terms of the elements of this C matrix, typically in the form of zero restrictions. The restriction $C_{i,j} = 0$ means that the (accumulated) response of the i-th variable to the j-th structural shock is zero in the long-run.

It is important to note that the expression for the long-run response (24.16) involves the inverse of A. Since EViews currently requires all restrictions to be linear in the elements of A and B, *if you specify a long-run restriction, the A matrix must be the identity matrix*.

To specify long-run restrictions by a pattern matrix, create a named matrix that contains the pattern for the long-run response matrix C. Unrestricted elements in the C matrix should be assigned a missing value "NA". For example, suppose you have a $k = 2$ variable VAR where you want to restrict the long-run response of the second endogenous vari-

able to the first structural shock to be zero $C_{2,1} = 0$. Then the long-run response matrix will have the following pattern:

$$C = \begin{pmatrix} \text{NA} & \text{NA} \\ 0 & \text{NA} \end{pmatrix} \tag{24.17}$$

You can create this matrix with the following commands:

```
matrix(2,2) patc = na
patc(2,1) = 0
```

Once you have created the pattern matrix, select **Proc/Estimate Structural Factorization...** from the VAR window menu. In the **SVAR Options** dialog, click the **Matrix** button and the **Long-Run Pattern** button and type in the name of the pattern matrix in the relevant edit box.

To specify the same long-run restriction in text form, select **Proc/Estimate Structural Factorization...** from the VAR window and click the **Text** button. In the edit window, you would type the following:

```
@lr2(@u1)=0 ' zero LR response of 2nd variable to 1st shock
```

where everything on the line after the apostrophe is a comment. This restriction begins with the special keyword "@LR#", with the "#" representing the response variable to restrict. Inside the parentheses, you must specify the impulse keyword "@U" and the innovation number, followed by an equal sign and the value of the response (typically 0). We caution you that while you can list multiple long-run restrictions, *you cannot mix short-run and long-run restrictions.*

Note that it is possible to specify long-run restrictions as short-run restrictions (by obtaining the infinite MA order representation). While the estimated A and B matrices should be the same, the impulse response standard errors from the short-run representation would be incorrect (since it does not take into account the uncertainty in the estimated infinite MA order coefficients).

Some Important Notes

Currently we have the following limitations for the specification of identifying restrictions:

- The A and B matrices must be square and non-singular. In text form, there must be exactly as many equations as there are endogenous variables in the VAR. For short-run restrictions in pattern form, you must provide the pattern matrices for both A and B matrices.

- The restrictions must be linear in the elements of A and B. Moreover, the restrictions on A and B must be independent (no restrictions across elements of A and B).
- You cannot impose both short-run and long-run restrictions.
- Structural decompositions are currently not available for VEC models.
- The identifying restriction assumes that the structural innovations u_t have unit variances. Therefore, you will almost always want to estimate the diagonal elements of the B matrix so that you obtain estimates of the standard deviations of the structural shocks.
- It is common in the literature to assume that the structural innovations have a diagonal covariance matrix rather than an identity matrix. To compare your results to those from these studies, you will have to divide each column of the B matrix with the diagonal element in that column (so that the resulting B matrix has ones on the main diagonal). To illustrate this transformation, consider a simple $k = 2$ variable model with $A = 1$:

$$\begin{aligned} e_{1,t} &= b_{11}u_{1,t} + b_{12}u_{2,t} \\ e_{2,t} &= b_{21}u_{1,t} + b_{22}u_{2,t} \end{aligned} \tag{24.18}$$

where $u_{1,t}$ and $u_{2,t}$ are independent structural shocks with unit variances as assumed in the EViews specification. To rewrite this specification with a B matrix containing ones on the main diagonal, define a new set of structural shocks by the transformations $v_{1,t} = b_{11}u_{1,t}$ and $v_{2,t} = b_{22}u_{2,t}$. Then the structural relation can be rewritten as,

$$\begin{aligned} e_{1,t} &= v_{1,t} + (b_{12}/b_{22})v_{2,t} \\ e_{2,t} &= (b_{21}/b_{11})v_{1,t} + v_{2,t} \end{aligned} \tag{24.19}$$

where now:

$$B = \begin{pmatrix} 1 & b_{12}/b_{22} \\ b_{21}/b_{11} & 1 \end{pmatrix}, \quad v_t = \begin{bmatrix} v_{1,t} \\ v_{2,t} \end{bmatrix} \sim \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} b_{11}^2 & 0 \\ 0 & b_{22}^2 \end{bmatrix} \right) \tag{24.20}$$

Note that the transformation involves only rescaling elements of the B matrix and not on the A matrix. For the case where B is a diagonal matrix, the elements in the main diagonal are simply the estimated standard deviations of the structural shocks.

Identification Conditions

As stated above, the assumption of orthonormal structural innovations imposes $k(k+1)/2$ restrictions on the $2k^2$ unknown elements in A and B, where k is the

number of endogenous variables in the VAR. In order to identify A and B, you need to provide at least $k(k+1)/2 - 2k^2 = k(3k-1)/2$ additional identifying restrictions. This is a necessary order condition for identification and is checked by counting the number of restrictions provided.

As discussed in Amisano and Giannini (1997), a sufficient condition for local identification can be checked by the invertibility of the "augmented" information matrix (see Amisano and Giannini, 1997). This local identification condition is evaluated numerically at the starting values. If EViews returns a singularity error message for different starting values, you should make certain that your restrictions identify the A and B matrices.

We also require the A and B matrices to be square and non-singular. The non-singularity condition is checked numerically at the starting values. If the A and B matrix is non-singular at the starting values, an error message will ask you to provide a different set of starting values.

Sign Indeterminacy

For some restrictions, the signs of the A and B matrices are not identified; see Christiano, Eichenbaum, and Evans (1999) for a discussion of this issue. When the sign is indeterminate, we choose a normalization so that the diagonal elements of the factorization matrix $A^{-1}B$ are all positive. This normalization ensures that all structural impulses have positive signs (as does the Cholesky factorization). The default is to always apply this normalization rules whenever applicable. If you do not want to switch the signs, deselect the **Normalize Sign** option from the **Optimization Control** tab of the **SVAR Options** dialog.

Estimation of *A* and *B* Matrices

Once you provide the identifying restrictions in any of the forms described above, you are ready to estimate the A and B matrices. Simply click the **OK** button in the **SVAR Options** dialog. You must first estimate these matrices in order to use the structural option in impulse responses and variance decompositions.

A and B are estimated by maximum likelihood, assuming the innovations are multivariate normal. We evaluate the likelihood in terms of unconstrained parameters by substituting out the constraints. The log likelihood is maximized by the method of scoring (with a Marquardt-type diagonal correction—See "Marquardt" on page 958), where the gradient and expected information matrix are evaluated analytically. See Amisano and Giannini (1997) for the analytic expression of these derivatives.

Optimization Control

Options for controlling the optimization process are provided in the **Optimization Control** tab of the **SVAR Options** dialog. You have the option to specify the starting values, maximum number of iterations, and the convergence criterion.

The starting values are those for the unconstrained parameters after substituting out the constraints. **Fixed** sets all free parameters to the value specified in the edit box. **User Specified** uses the values in the coefficient vector as specified in text form as starting values. For restrictions specified in pattern form, user specified starting values are taken from the first m elements of the default C coefficient vector, where m is the number of free parameters. **Draw from...** options randomly draw the starting values for the free parameters from the specified distributions.

Estimation Output

Once convergence is achieved, EViews displays the estimation output in the VAR window. The point estimates, standard errors, and z-statistics of the estimated free parameters are reported together with the maximized value of the log likelihood. The estimated standard errors are based on the inverse of the estimated information matrix (negative expected value of the Hessian) evaluated at the final estimates.

For overidentified models, we also report the LR test for over-identification. The LR test statistic is computed as:

$$LR = 2(l_u - l_r) = T(\text{tr}(P) - \log|P| - k) \tag{24.21}$$

where $P = A'B^{-T}B^{-1}A\Sigma$. Under the null hypothesis that the restrictions are valid, the LR statistic is asymptotically distributed $\chi^2(q-k)$ where q is the number of identifying restrictions.

If you switch the view of the VAR window, you can come back to the previous results (without reestimating) by selecting **View/SVAR Output** from the VAR window. In addition, some of the SVAR estimation results can be retrieved as data members of the VAR; see "Var Data Members" on page 191 of the *Command and Programming Reference* for a list of available VAR data members.

Cointegration Test

The finding that many macro time series may contain a unit root has spurred the development of the theory of non-stationary time series analysis. Engle and Granger (1987) pointed out that a linear combination of two or more non-stationary series may be stationary. If such a stationary linear combination exists, the non-stationary time series are said to be *cointegrated*. The stationary linear combination is called the *cointegrating equation* and may be interpreted as a long-run equilibrium relationship among the variables.

The purpose of the cointegration test is to determine whether a group of non-stationary series are cointegrated or not. As explained below, the presence of a cointegrating relation forms the basis of the VEC specification. EViews implements VAR-based cointegration tests using the methodology developed in Johansen (1991, 1995a).

Consider a VAR of order p:

$$y_t = A_1 y_{t-1} + \cdots + A_p y_{t-p} + Bx_t + \epsilon_t \tag{24.22}$$

where y_t is a k-vector of non-stationary I(1) variables, x_t is a d-vector of deterministic variables, and ϵ_t is a vector of innovations. We may rewrite this VAR as,

$$\Delta y_t = \Pi y_{t-1} + \sum_{i=1}^{p-1} \Gamma_i \Delta y_{t-i} + Bx_t + \epsilon_t \tag{24.23}$$

where:

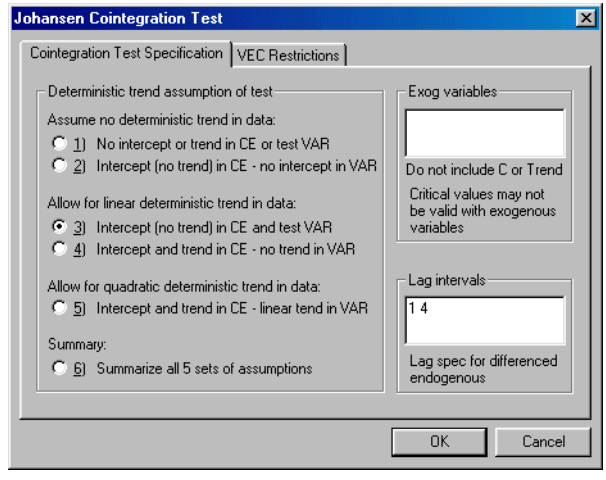
$$\Pi = \sum_{i=1}^{p} A_i - I, \qquad \Gamma_i = -\sum_{j=i+1}^{p} A_j \tag{24.24}$$

Granger's representation theorem asserts that if the coefficient matrix Π has reduced rank $r < k$, then there exist $k \times r$ matrices α and β each with rank r such that $\Pi = \alpha\beta'$ and $\beta' y_t$ is I(0). r is the number of cointegrating relations (the *cointegrating rank*) and each column of β is the cointegrating vector. As explained below, the elements of α are known as the adjustment parameters in the VEC model. Johansen's method is to estimate the Π matrix from an unrestricted VAR and to test whether we can reject the restrictions implied by the reduced rank of Π.

How to Perform a Cointegration Test

To carry out the Johansen cointegration test, select **View/Cointegration Test...** from the group or VAR window toolbar. The **Cointegration Test Specification** page prompts you for information about the test.

Note that since this is a test for cointegration, this test is only valid when you are working with series that are known to be nonstationary. You may wish first to apply unit root tests to each series in the VAR. See "Unit Root Test" on page 329 for details on carrying out unit root tests in EViews.


Johansen Cointegration Test
Cointegration Test Specification
VEC Restrictions
Deterministic trend assumption of test
Assume no deterministic trend in data:
1) No intercept or trend in CE or test VAR
2) Intercept (no trend) in CE - no intercept in VAR
Allow for linear deterministic trend in data:
3) Intercept (no trend) in CE and test VAR
4) Intercept and trend in CE - no trend in VAR
Allow for quadratic deterministic trend in data:
5) Intercept and trend in CE - linear tend in VAR
Summary:
6) Summarize all 5 sets of assumptions
Exog variables
Do not include C or Trend
Critical values may not be valid with exogenous variables
Lag intervals
1 4
Lag spec for differenced endogenous
OK
Cancel

Deterministic Trend Specification

Your series may have nonzero means and deterministic trends as well as stochastic trends. Similarly, the cointegrating

equations may have intercepts and deterministic trends. The asymptotic distribution of the LR test statistic for cointegration does not have the usual χ^2 distribution and depends on the assumptions made with respect to deterministic trends. Therefore, in order to carry out the test, you need to make an assumption regarding the trend underlying your data.

For each row case in the dialog, the COINTEQ column lists the deterministic variables that appear inside the cointegrating relations (error correction term), while the OUTSIDE column lists the deterministic variables that appear in the VEC equation outside the cointegrating relations. Cases 2 and 4 do not have the same set of deterministic terms in the two columns. For these two cases, some of the deterministic term is restricted to belong only in the cointegrating relation. For cases 3 and 5, the deterministic terms are common in the two columns and the decomposition of the deterministic effects inside and outside the cointegrating space is not uniquely identified; see the technical discussion below.

In practice, cases 1 and 5 are rarely used. You should use case 1 only if you know that all series have zero mean. Case 5 may provide a good fit in-sample but will produce implausible forecasts out-of-sample. As a rough guide, use case 2 if none of the series appear to have a trend. For trending series, use case 3 if you believe all trends are stochastic; if you believe some of the series are trend stationary, use case 4.

If you are not certain which trend assumption to use, you may choose the **Summary of all 5 trend assumptions** option (case 6) to help you determine the choice of the trend assumption. This option indicates the number of cointegrating relations under each of the 5 trend assumptions, and you will be able to assess the sensitivity of the results to the trend assumption.

Technical Discussion

EViews considers the following five deterministic trend cases considered by Johansen (1995a, pp. 80–84):

1. The level data y_t have no deterministic trends and the cointegrating equations do not have intercepts:

$$H_2(r)\colon\ \Pi y_{t-1} + Bx_t = \alpha\beta' y_{t-1}$$

2. The level data y_t have no deterministic trends and the cointegrating equations have intercepts:

$$H_1^*(r)\colon\ \Pi y_{t-1} + Bx_t = \alpha(\beta' y_{t-1} + \rho_0)$$

3. The level data y_t have linear trends but the cointegrating equations have only intercepts:

$$H_1(r)\colon\ \Pi y_{t-1} + Bx_t = \alpha(\beta' y_{t-1} + \rho_0) + \alpha_\perp \gamma_0$$

4. The level data y_t and the cointegrating equations have linear trends:

$$H^*(r)\text{: } \Pi y_{t-1} + Bx_t = \alpha(\beta' y_{t-1} + \rho_0 + \rho_1 t) + \alpha_\perp \gamma_0$$

5. The level data y_t have quadratic trends and the cointegrating equations have linear trends:

$$H(r)\text{: } \Pi y_{t-1} + Bx_t = \alpha(\beta' y_{t-1} + \rho_0 + \rho_1 t) + \alpha_\perp(\gamma_0 + \gamma_1 t)$$

The terms associated with $\alpha_\perp$ are the deterministic terms "outside" the cointegrating relations. When a deterministic term appears both inside and outside the cointegrating relation, the decomposition is not uniquely identified. Johansen (1995a) identifies the part that belongs inside the error correction term by orthogonally projecting the exogenous terms onto the α space so that $\alpha_\perp$ is the null space of α such that $\alpha' \alpha_\perp = 0$. EViews uses a different identification method so that the error correction term has a sample mean of zero. More specifically, we identify the part inside the error correction term by regressing the cointegrating relations $\beta' y_t$ on a constant (and linear trend).

Exogenous Variables

The test dialog allows you to specify additional exogenous variables x_t to include in the test VAR. *The constant and linear trend should not be listed in the edit box* since they are specified using the five **Trend Specification** options. If you choose to include exogenous variables, be aware that the critical values reported by EViews *do not account* for these variables.

The most commonly added deterministic terms are seasonal dummy variables. Note, however, that if you include standard 0–1 seasonal dummy variables in the test VAR, this will affect both the mean and the trend of the level series y_t. To handle this problem, Johansen (1995a, page 84) suggests using centered (orthogonalized) seasonal dummy variables, which shift the mean without contributing to the trend. Centered seasonal dummy variables for quarterly and monthly series can be generated by the commands:

```
series d_q = @seas(q) - 1/4
series d_m = @seas(m) - 1/12
```

for quarter q and month m, respectively.

Lag Intervals

You should specify the lags of the test VAR as pairs of intervals. Note that the lags are specified as *lags of the first differenced terms* used in the auxiliary regression, not in terms of the levels. For example, if you type "`1 2`" in the edit field, the test VAR regresses Δy_t on Δy_{t-1}, Δy_{t-2}, and any other exogenous variables that you have specified. Note that in terms of the level series y_t the largest lag is 3. To run a cointegration test with one lag in the *level* series, type "`0 0`" in the edit field.

Interpreting Results of a Cointegration Test

As an example, the first part of the cointegration test output for the four-variable system used by Johansen and Juselius (1990) for the Danish data is shown below.

Date: 01/16/04 Time: 11:40
Sample (adjusted): 1974:3 1987:3
Included observations: 53 after adjusting endpoints
Trend assumption: No deterministic trend (restricted constant)
Series: LRM LRY IBO IDE
Lags interval (in first differences): 1 to 1

Unrestricted Cointegration Rank Test (Trace)

Hypothesized No. of CE(s)	Eigenvalue	Trace Statistic	0.05 Critical Value	Prob.**
None	0.469677	52.71087	54.0790	0.0659
At most 1	0.174241	19.09464	35.1928	0.7814
At most 2	0.118083	8.947661	20.2618	0.7411
At most 3	0.042249	2.287849	9.1645	0.7200

* denotes rejection of the hypothesis at the 0.05 level
Trace test indicates no cointegration at the 0.05 level
**MacKinnon-Haug-Michelis (1999) p-values

As indicated in the header of the output, the test assumes no trend in the series with a restricted intercept in the cointegration relation (second trend specification in the dialog), includes three orthogonalized seasonal dummy variables D1–D3, and uses one lag in differences (two lags in levels) which is specified as "`1 1`" in the edit field.

Number of Cointegrating Relations

The first part of the table reports results for testing the number of cointegrating relations. Two types of test statistics are reported. The first block reports the so-called *trace* statistics and the second block (not shown above) reports the *maximum eigenvalue* statistics. For each block, the first column is the number of cointegrating relations under the null hypothesis, the second column is the ordered eigenvalues of the Π matrix in (24.24), the third column is the test statistic, and the last two columns are the 5% and 1% critical values. The (nonstandard) critical values are taken from Osterwald-Lenum (1992), which differ slightly from those reported in Johansen and Juselius (1990).

To determine the number of cointegrating relations r *conditional on the assumptions made about the trend*, we can proceed sequentially from $r = 0$ to $r = k - 1$ until we fail to reject. The result of this sequential testing procedure is reported at the bottom of each table block.

The trace statistic reported in the first block tests the null hypothesis of r cointegrating relations against the alternative of k cointegrating relations, where k is the number of endogenous variables, for $r = 0, 1, \ldots, k-1$. The alternative of k cointegrating relations corresponds to the case where none of the series has a unit root and a stationary VAR may be specified in terms of the levels of all of the series. The trace statistic for the null hypothesis of r cointegrating relations is computed as:

$$LR_{\text{tr}}(r|k) = -T \sum_{i=r+1}^{k} \log(1-\lambda_i) \tag{24.25}$$

where λ_i is the i-th largest eigenvalue of the Π matrix in (24.24) which is reported in the second column of the output table.

The second block of the output reports the maximum eigenvalue statistic which tests the null hypothesis of r cointegrating relations against the alternative of $r+1$ cointegrating relations. This test statistic is computed as:

$$\begin{aligned} LR_{\text{max}}(r|r+1) &= -T\log(1-\lambda_{r+1}) \\ &= LR_{\text{tr}}(r|k) - LR_{\text{tr}}(r+1|k) \end{aligned} \tag{24.26}$$

for $r = 0, 1, \ldots, k-1$.

There are a few other details to keep in mind:

- Critical values are available for up to $k = 10$ series. Also note that the critical values depend on the trend assumptions and may not be appropriate for models that contain other deterministic regressors. For example, a shift dummy variable in the test VAR implies a broken linear trend in the level series y_t.
- The trace statistic and the maximum eigenvalue statistic may yield conflicting results. For such cases, we recommend that you examine the estimated cointegrating vector and base your choice on the interpretability of the cointegrating relations; see Johansen and Juselius (1990) for an example.
- In some cases, the individual unit root tests will show that some of the series are integrated, but the cointegration test will indicate that the Π matrix has full rank ($r = k$). This apparent contradiction may be the result of low power of the cointegration tests, stemming perhaps from a small sample size or serving as an indication of specification error.

Cointegrating relations

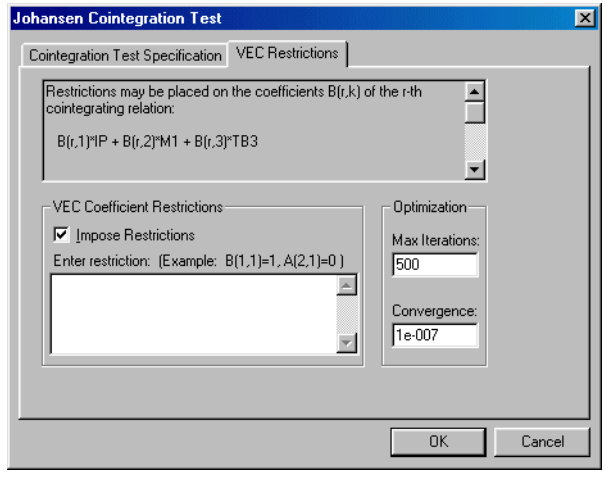
The second part of the output provides estimates of the cointegrating relations β and the adjustment parameters α. As is well known, the cointegrating vector β is not identified unless we impose some arbitrary normalization. The first block reports estimates of β and α based on the normalization $\beta' S_{11}\beta = I$, where S_{11} is defined in Johansen (1995a).

Note that the *transpose* of β is reported under **Unrestricted Cointegrating Coefficients** so that the first *row* is the first cointegrating vector, the second *row* is the second cointegrating vector, and so on.

The remaining blocks report estimates from a different normalization for each possible number of cointegrating relations $r = 0, 1, \ldots, k-1$. This alternative normalization expresses the first r variables as functions of the remaining $k-r$ variables in the system. Asymptotic standard errors are reported in parentheses for the parameters that are identified.

Imposing Restrictions

Since the cointegrating vector β is not identified, you may wish to impose your own identifying restrictions. Restrictions can be imposed on the cointegrating vector (elements of the β matrix) and/or on the adjustment coefficients (elements of the α matrix). To impose restrictions in a cointegration test, select **View/Cointegration Test...** and specify the options in the **Trend Specification** tab as explained above. Then bring up the **VEC Restrictions** tab. You will enter your restrictions in the edit box that appears when you check the **Impose Restrictions** box:



Restrictions on the Cointegrating Vector

To impose restrictions on the cointegrating vector β, you must refer to the (i,j)-th element of the *transpose* of the β matrix by `B(i,j)`. The i-th cointegrating relation has the representation:

```
B(i,1)*y1 + B(i,2)*y2 + ... + B(i,k)*yk
```

where `y1`, `y2`, ... are the (lagged) endogenous variable. Then, if you want to impose the restriction that the coefficient on `y1` for the second cointegrating equation is 1, you would type the following in the edit box:

```
B(2,1) = 1
```

You can impose multiple restrictions by separating each restriction with a comma on the same line or typing each restriction on a separate line. For example, if you want to impose the restriction that the coefficients on `y1` for the first and second cointegrating equations are 1, you would type:

```
B(1,1) = 1
B(2,1) = 1
```

Currently *all restrictions must be linear (or more precisely affine) in the elements of the β matrix*. So for example

```
B(1,1) * B(2,1) = 1
```

will return a syntax error.

Restrictions on the Adjustment Coefficients

To impose restrictions on the adjustment coefficients, you must refer to the (i,j)-th elements of the α matrix by `A(i,j)`. The error correction terms in the i-th VEC equation will have the representation:

```
A(i,1)*CointEq1 + A(i,2)*CointEq2 + ... + A(i,r)*CointEqr
```

Restrictions on the adjustment coefficients are currently limited to linear homogeneous restrictions so that you must be able to write your restriction as $R\text{vec}(\alpha) = 0$, where R is a known $qk \times r$ matrix. This condition implies, for example, that the restriction,

```
A(1,1) = A(2,1)
```

is valid but:

```
A(1,1) = 1
```

will return a restriction syntax error.

One restriction of particular interest is whether the i-th row of the α matrix is all zero. If this is the case, then the i-th endogenous variable is said to be *weakly exogenous with respect to the β parameters*. See Johansen (1992b) for the definition and implications of weak exogeneity. For example, if we assume that there is only one cointegrating relation in the VEC, to test whether the second endogenous variable is weakly exogenous with respect to β you would enter:

```
A(2,1) = 0
```

To impose multiple restrictions, you may either separate each restriction with a comma on the same line or type each restriction on a separate line. For example, to test whether the second endogenous variable is weakly exogenous with respect to β in a VEC with two cointegrating relations, you can type:

```
A(2,1) = 0
A(2,2) = 0
```

You may also impose restrictions on both β *and* α. However, the restrictions on β and α must be *independent*. So for example,

```
A(1,1) = 0
B(1,1) = 1
```

is a valid restriction but:

```
A(1,1) = B(1,1)
```

will return a restriction syntax error.

Identifying Restrictions and Binding Restrictions

EViews will check to see whether the restrictions you provided identify all cointegrating vectors for each possible rank. The identification condition is checked numerically by the rank of the appropriate Jacobian matrix; see Boswijk (1995) for the technical details. Asymptotic standard errors for the estimated cointegrating parameters will be reported only if the restrictions identify the cointegrating vectors.

If the restrictions are binding, EViews will report the LR statistic to test the binding restrictions. The LR statistic is reported if the degrees of freedom of the asymptotic χ^2 distribution is positive. Note that the restrictions can be binding even if they are not identifying, (*e.g.* when you impose restrictions on the adjustment coefficients but not on the cointegrating vector).

Options for Restricted Estimation

Estimation of the restricted cointegrating vectors β and adjustment coefficients α generally involves an iterative process. The **VEC Restrictions** tab provides iteration control for the maximum number of iterations and the convergence criterion. EViews estimates the restricted β and α using the switching algorithm as described in Boswijk (1995). Each step of the algorithm is guaranteed to increase the likelihood and the algorithm should eventually converge (though convergence may be to a local rather than a global optimum). You may need to increase the number of iterations in case you are having difficulty achieving convergence at the default settings.

Results of Restricted Cointegration Test

If you impose restrictions in the **Cointegration Test** view, the output will first display the test results without the restrictions as described above. The second part of the output begins by displaying the results of the LR test for binding restrictions.

Restrictions:

a(3,1)=0

Tests of cointegration restrictions:

Hypothesized No. of CE(s)	Restricted Log-likehood	LR Statistic	Degrees of Freedom	Probability
1	668.6698	0.891088	1	0.345183
2	674.2964	NA	NA	NA
3	677.4677	NA	NA	NA

NA indicates restriction not binding.

If the restrictions are not binding for a particular rank, the corresponding rows will be filled with NAs. If the restrictions are binding but the algorithm did not converge, the corresponding row will be filled with an asterisk "*". (You should redo the test by increasing the number of iterations or relaxing the convergence criterion.) For the example output displayed above, we see that the single restriction $\alpha_{31} = 0$ is binding only under the assumption that there is one cointegrating relation. *Conditional on there being only one cointegrating relation*, the LR test does not reject the imposed restriction at conventional levels.

The output also reports the estimated β and α imposing the restrictions. Since the cointegration test does not specify the number of cointegrating relations, results for all ranks that are consistent with the specified restrictions will be displayed. For example, suppose the restriction is:

```
B(2,1) = 1
```

Since this is a restriction on the second cointegrating vector, EViews will display results for ranks $r = 2, 3, \ldots, k-1$ (if the VAR has only $k = 2$ variables, EViews will return an error message pointing out that the "implied rank from restrictions must be of reduced order").

For each rank, the output reports whether convergence was achieved and the number of iterations. The output also reports whether the restrictions identify all cointegrating parameters under the assumed rank. If the cointegrating vectors are identified, asymptotic standard errors will be reported together with the parameters β.

Vector Error Correction (VEC) Models

A vector error correction (VEC) model is a restricted VAR designed for use with nonstationary series that are known to be cointegrated. The VEC has cointegration relations built into the specification so that it restricts the long-run behavior of the endogenous variables to converge to their cointegrating relationships while allowing for short-run adjustment dynamics. The cointegration term is known as the *error correction* term since the deviation from long-run equilibrium is corrected gradually through a series of partial short-run adjustments.

To take the simplest possible example, consider a two variable system with one cointegrating equation and no lagged difference terms. The cointegrating equation is:

$$y_{2,t} = \beta y_{1,t} \tag{24.27}$$

The corresponding VEC model is:

$$\begin{aligned} \Delta y_{1,t} &= \alpha_1(y_{2,t-1} - \beta y_{1,t-1}) + \epsilon_{1,t} \\ \Delta y_{2,t} &= \alpha_2(y_{2,t-1} - \beta y_{1,t-1}) + \epsilon_{2,t} \end{aligned} \tag{24.28}$$

In this simple model, the only right-hand side variable is the error correction term. In long run equilibrium, this term is zero. However, if y_1 and y_2 deviate from the long run equilibrium, the error correction term will be nonzero and each variable adjusts to partially restore the equilibrium relation. The coefficient α_i measures the speed of adjustment of the i-th endogenous variable towards the equilibrium.

How to Estimate a VEC

As the VEC specification only applies to cointegrated series, you should first run the Johansen cointegration test as described above and determine the number of cointegrating relations. You will need to provide this information as part of the VEC specification.

To set up a VEC, click the **Estimate** button in the VAR toolbar and choose the **Vector Error Correction** specification from the **VAR/VEC Specification** tab. In the **VAR/VEC Specification** tab, you should provide the same information as for an unrestricted VAR, except that:

- The constant or linear trend term should *not* be included in the **Exogenous Series** edit box. The constant and trend specification for VECs should be specified in the **Cointegration** tab (see below).
- The lag interval specification refers to *lags of the first difference terms* in the VEC. For example, the lag specification "`1 1`" will include lagged first difference terms on the right-hand side of the VEC. Rewritten in levels, this VEC is a restricted VAR with two lags. To estimate a VEC with no lagged first difference terms, specify the lag as "`0 0`".

- The constant and trend specification for VECs should be specified in the **Cointegration** tab. You must choose from one of the five trend specifications as explained in "Deterministic Trend Specification" on page 740. You must also specify the number of cointegrating relations in the appropriate edit field. This number should be a positive integer less than the number of endogenous variables in the VEC.
- If you want to impose restrictions on the cointegrating relations and/or the adjustment coefficients, use the **Restrictions** tab. "Restrictions on the Cointegrating Vector" on page 745 describes this restriction in greater detail. Note that this tab is grayed out unless you have clicked the **Vector Error Correction** specification in the **VAR/VEC Specification** tab.

Once you have filled the dialog, simply click **OK** to estimate the VEC. Estimation of a VEC model is carried out in two steps. In the first step, we estimate the cointegrating relations from the Johansen procedure as used in the cointegration test. We then construct the error correction terms from the estimated cointegrating relations and estimate a VAR in first differences including the error correction terms as regressors.

VEC Estimation Output

The VEC estimation output consists of two parts. The first part reports the results from the first step Johansen procedure. If you did not impose restrictions, EViews will use a default normalization that identifies all cointegrating relations. This default normalization expresses the first r variables in the VEC as functions of the remaining $k-r$ variables, where r is the number of cointegrating relations and k is the number of endogenous variables. Asymptotic standard errors (corrected for degrees of freedom) are reported for parameters that are identified under the restrictions. If you provided your own restrictions, standard errors will not be reported unless the restrictions identify all cointegrating vectors.

The second part of the output reports results from the second step VAR in first differences, including the error correction terms estimated from the first step. The error correction terms are denoted `CointEq1`, `CointEq2`, and so on in the output. This part of the output has the same format as the output from unrestricted VARs as explained in "VAR Estimation Output" on page 723, with one difference. At the bottom of the VEC output table, you will see two log likelihood values reported for the system. The first value, labeled **Log Likelihood (d.f. adjusted),** is computed using the determinant of the residual covariance matrix (reported as **Determinant Residual Covariance**), using small sample degrees of freedom correction as in (24.3). This is the log likelihood value reported for unrestricted VARs. The **Log Likelihood** value is computed using the residual covariance matrix without correcting for degrees of freedom. This log likelihood value is comparable to the one reported in the cointegration test output.

Views and Procs of a VEC

Views and procs available for VECs are mostly the same as those available for VARs as explained above. Here, we only mention those that are specific to VECs.

Cointegrating Relations

View/Cointegration Graph displays a graph of the estimated cointegrating relations as used in the VEC. To store these estimated cointegrating relations as named series in the workfile, use **Proc/Make Cointegration Group**. This proc will create and display an untitled group object containing the estimated cointegrating relations as named series. These series are named COINTEQ01, COINTEQ02 and so on.

Forecasting

Currently forecasts from a VAR or VEC are not available from the VAR object. Forecasts can be obtained by solving a model created from the estimated VAR/VEC. Click on **Proc/Make Model** from the VAR window toolbar to create a model object from the estimated VAR/VEC. You may then make any changes to the model specification, including modifying the ASSIGN statement before solving the model to obtain the forecasts. See Chapter 26, "Models", on page 777, for further discussion on how to forecast from model objects in EViews.

Data Members

Various results from the estimated VAR/VEC can be retrieved through the command line data members. "Var Data Members" on page 191 of the *Command and Programming Reference* provides a complete list of data members that are available for a VAR object. Here, we focus on retrieving the estimated coefficients of a VAR/VEC.

Obtaining Coefficients of a VAR

Coefficients of (unrestricted) VARs can be accessed by referring to elements of a *two dimensional array* C. The first dimension of C refers to the equation number of the VAR, while the second dimension refers to the variable number in each equation. For example, C(2,3) is the coefficient of the third regressor in the second equation of the VAR. The C(2,3) coefficient of a VAR named VAR01 can then be accessed by the command

```
var01.c(2,3)
```

To examine the correspondence between each element of C and the estimated coefficients, select **View/Representations** from the VAR toolbar.

Obtaining Coefficients of a VEC

For VEC models, the estimated coefficients are stored in three different two dimensional arrays: A, B, and C. A contains the adjustment parameters α, B contains the cointegrating

vectors β', and C holds the short-run parameters (the coefficients on the lagged first difference terms).

- The first index of A is the equation number of the VEC, while the second index is the number of the cointegrating equation. For example, A(2,1) is the adjustment coefficient of the first cointegrating equation in the second equation of the VEC.
- The first index of B is the number of the cointegrating equation, while the second index is the variable number in the cointegrating equation. For example, B(2,1) is the coefficient of the first variable in the second cointegrating equation. Note that this indexing scheme corresponds to the *transpose* of β.
- The first index of C is the equation number of the VEC, while the second index is the variable number of the first differenced regressor of the VEC. For example, C(2, 1) is the coefficient of the first differenced regressor in the second equation of the VEC.

You can access each element of these coefficients by referring to the name of the VEC followed by a dot and coefficient element:

```
var01.a(2,1)
var01.b(2,1)
var01.c(2,1)
```

To see the correspondence between each element of A, B, and C and the estimated coefficients, select **View/Representations** from the VAR toolbar.

A Note on Version Compatibility

The following changes made in Version 4 may yield VAR results that do not match those reported from previous versions of EViews:

- The estimated residual covariance matrix is now computed using the finite sample adjustment so the sum-of-squares is divided by $T-p$ where p is the number of estimated coefficients in each VAR equation. Previous versions of EViews divided the sum-of-squares by T.
- The standard errors for the cointegrating vector are now computed using the more general formula in Boswijk (1995), which also covers the restricted case.

Chapter 25. State Space Models and the Kalman Filter

The EViews sspace (state space) object provides a straightforward, easy-to-use interface for specifying, estimating, and working with the results of your single or multiple equation dynamic system. EViews provides a wide range of specification, filtering, smoothing, and other forecasting tools which aid you in working with dynamic systems specified in state space form.

A wide range of time series models, including the classical linear regression model and ARIMA models, can be written and estimated as special cases of a state space specification. State space models have been applied in the econometrics literature to model unobserved variables: (rational) expectations, measurement errors, missing observations, permanent income, unobserved components (cycles and trends), and the non-accelerating rate of unemployment. Extensive surveys of applications of state space models in econometrics can be found in Hamilton (1994a, Chapter 13; 1994b) and Harvey (1989, Chapters 3, 4).

There are two main benefits to representing a dynamic system in state space form. First, the state space allows unobserved variables (known as the state variables) to be incorporated into, and estimated along with, the observable model. Second, state space models can be analyzed using a powerful recursive algorithm known as the Kalman (Bucy) filter. The Kalman filter algorithm has been used, among other things, to compute exact, finite sample forecasts for Gaussian ARMA models, multivariate (vector) ARMA models, MIMIC (multiple indicators and multiple causes), Markov switching models, and time varying (random) coefficient models.

Those of you who have used early versions of the sspace object will note that much was changed with the EViews 4 release. We strongly recommend that you read "Converting from Version 3 Sspace" on page 775 before loading existing workfiles and before beginning to work with the new state space routines.

Background

We present here a very brief discussion of the specification and estimation of a linear state space model. Those desiring greater detail are directed to Harvey (1989), Hamilton (1994a, Chapter 13, 1994b), and especially the excellent treatment of Koopman, Shephard and Doornik (1999).

Specification

A linear state space representation of the dynamics of the $n \times 1$ vector y_t is given by the system of equations:

$$y_t = c_t + Z_t\alpha_t + \epsilon_t \tag{25.1}$$

$$\alpha_{t+1} = d_t + T_t\alpha_t + v_t \tag{25.2}$$

where α_t is an $m \times 1$ vector of possibly unobserved state variables, where c_t, Z_t, d_t and T_t are conformable vectors and matrices, and where ϵ_t and v_t are vectors of mean zero, Gaussian disturbances. Note that the unobserved state vector is assumed to move over time as a first-order vector autoregression.

We will refer to the first set of equations as the "signal" or "observation" equations and the second set as the "state" or "transition" equations. The disturbance vectors ϵ_t and v_t are assumed to be serially independent, with contemporaneous variance structure:

$$\Omega_t = \operatorname{var}\begin{bmatrix}\epsilon_t \\ v_t\end{bmatrix} = \begin{bmatrix}H_t & G_t \\ G_t' & Q_t\end{bmatrix} \tag{25.3}$$

where H_t is an $n \times n$ symmetric variance matrix, Q_t is an $m \times m$ symmetric variance matrix, and G_t is an $n \times m$ matrix of covariances.

In the discussion that follows, we will generalize the specification given in (25.1)—(25.3) by allowing the system matrices and vectors $\Xi_t \equiv \{c_t, d_t, Z_t, T_t, H_t, Q_t, G_t\}$ to depend upon observable explanatory variables X_t and unobservable parameters θ. Estimation of the parameters θ is discussed in "Estimation" beginning on page 757.

Filtering

Consider the conditional distribution of the state vector α_t given information available at time s. We can define the mean and variance matrix of the conditional distribution as:

$$a_{t|s} \equiv E_s(\alpha_t) \tag{25.4}$$

$$P_{t|s} \equiv E_s[(\alpha_t - a_{t|s})(\alpha_t - a_{t|s})'] \tag{25.5}$$

where the subscript below the expectation operator indicates that expectations are taken using the conditional distribution for that period.

One important conditional distribution is obtained by setting $s = t-1$, so that we obtain the *one-step ahead mean* $a_{t|t-1}$ and *one-step ahead variance* $P_{t|t-1}$ of the states α_t. Under the Gaussian error assumption, $a_{t|t-1}$ is also the minimum mean square error estimator of α_t and $P_{t|t-1}$ is the mean square error (MSE) of $a_{t|t-1}$. If the normality assumption is dropped, $a_{t|t-1}$ is still the minimum mean square *linear* estimator of α_t.

Given the one-step ahead state conditional mean, we can also form the (linear) minimum MSE *one-step ahead estimate* of y_t:

$$\tilde{y}_t = y_{t|t-1} \equiv E_{t-1}(y_t) = E(y_t | a_{t|t-1}) = c_t + Z_t a_{t|t-1} \tag{25.6}$$

The *one-step ahead prediction error* is given by,

$$\tilde{\epsilon}_t \;=\; \epsilon_{t|t-1} \equiv y_t - \tilde{y}_{t|t-1} \tag{25.7}$$

and the *prediction error variance* is defined as:

$$\tilde{F}_t \;=\; F_{t|t-1} \equiv \text{var}(\epsilon_{t|t-1}) \;=\; Z_t P_{t|t-1} Z_t' + H_t \tag{25.8}$$

The Kalman (Bucy) filter is a recursive algorithm for sequentially updating the one-step ahead estimate of the state mean and variance given new information. Details on the recursion are provided in the references above. For our purposes, it is sufficient to note that given initial values for the state mean and covariance, values for the system matrices Ξ_t, and observations on y_t, the Kalman filter may be used to compute one-step ahead estimates of the state and the associated mean square error matrix, $\{a_{t|t-1}, P_{t|t-1}\}$, the contemporaneous or *filtered* state mean and variance, $\{a_t, P_t\}$, and the one-step ahead prediction, prediction error, and prediction error variance, $\{y_{t|t-1}, \epsilon_{t|t-1}, F_{t|t-1}\}$. Note that we may also obtain the standardized prediction residual, $e_{t|t-1}$, by dividing $\epsilon_{t|t-1}$ by the square-root of the corresponding diagonal element of $F_{t|t-1}$.

Fixed-Interval Smoothing

Suppose that we observe the sequence of data up to time period T. The process of using this information to form expectations at any time period up to T is known as *fixed-interval smoothing*. Despite the fact that there are a variety of other distinct forms of smoothing (*e.g.*, fixed-point, fixed-lag), we will use the term *smoothing* to refer to fixed-interval smoothing.

Additional details on the smoothing procedure are provided in the references given above. For now, note that smoothing uses all of the information in the sample to provide *smoothed estimates of the states*, $\hat{\alpha}_t \equiv a_{t|T} \equiv \text{E}_T(\alpha_t)$, and *smoothed estimates of the state variances*, $V_t \equiv \text{var}_T(\alpha_t)$. The matrix V_t may also be interpreted as the MSE of the smoothed state estimate $\hat{\alpha}_t$.

As with the one-step ahead states and variances above, we may use the smoothed values to form s*moothed estimates of the signal variables*,

$$\hat{y}_t \equiv E(y_t | \hat{\alpha}_t) \;=\; c_t + Z_t \hat{\alpha}_t \tag{25.9}$$

and to compute the *variance of the smoothed signal estimates:*

$$S_t \equiv \text{var}(\hat{y}_{t|T}) \;=\; Z_t V_t Z_t' \,. \tag{25.10}$$

Lastly, the smoothing procedure allows us to compute *smoothed disturbance estimates*, $\hat{\epsilon}_t \equiv \epsilon_{t|T} \equiv E_T(\epsilon_t)$ and $\hat{v}_t \equiv v_{t|T} \equiv E_T(v_t)$, and a corresponding *smoothed disturbance variance matrix*:

$$\hat{\Omega}_t = \text{var}_T\left(\begin{bmatrix}\epsilon_t \\ v_t\end{bmatrix}\right) \tag{25.11}$$

Dividing the smoothed disturbance estimates by the square roots of the corresponding diagonal elements of the smoothed variance matrix yields the *standardized smoothed disturbance estimates* $\hat{e}_t$ and $\hat{\nu}_t$.

Forecasting

There are a variety of types of forecasting which may be performed with state space models. These methods differ primarily in what and how information is used. We will focus on the three methods that are supported by EViews built-in forecasting routines.

n-Step Ahead Forecasting

Earlier, we examined the notion of one-step ahead prediction. Consider now the notion of multi-step ahead prediction of observations, in which we take a fixed set of information available at a given period, and forecast several periods ahead. Modifying slightly the expressions in (25.4)—(25.8) yields the *n-step ahead state conditional mean and variance:*

$$a_{t+n|t} \equiv E_t(\alpha_{t+n}) , \tag{25.12}$$

$$P_{t+n|t} \equiv E_t[(\alpha_{t+n} - a_{t+n|t})(\alpha_{t+n} - a_{t+n|t})'] \tag{25.13}$$

the *n-step ahead forecast,*

$$y_{t+n|t} \equiv E_t(y_{t+n}) = c_t + Z_t a_{t+n|t} \tag{25.14}$$

and the corresponding *n-step ahead forecast MSE matrix:*

$$F_{t+n|t} \equiv \text{MSE}(\tilde{y}_{t+n|t}) = Z_{t+n} P_{t+n|t} Z_{t+n}' + H_t \tag{25.15}$$

for $n = 1, 2, \ldots$. As before, $a_{t+n|t}$ may also be interpreted as the minimum MSE estimate of α_{t+n} based on the information set available at time t, and $P_{t+n|t}$ is the MSE of the estimate.

It is worth emphasizing that the definitions given above for the forecast MSE matrices $F_{t+n|t}$ do not account for extra variability introduced in the estimation of any unknown parameters θ. In this setting, the $F_{t+n|t}$ will understate the true variability of the forecast, and should be viewed as being computed conditional on the specific value of the estimated parameters.

It is also worth noting that the n-step ahead forecasts may be computed using a slightly modified version of the basic Kalman recursion (Harvey 1989). To forecast at period $s = t + n$, simply initialize a Kalman filter at time $t + 1$ with the values of the predicted states and state covariances using information at time t, and run the filter forward

$n-1$ additional periods using no additional signal information. This procedure is repeated for each observation in the forecast sample, $s = t+1, \ldots, t+n^*$.

Dynamic Forecasting

The concept of *dynamic forecasting* should be familiar to you from other EViews estimation objects. In dynamic forecasting, we start at the beginning of the forecast sample t, and compute a complete *set* of n-period ahead forecasts for each period $n = 1, \ldots, n^*$ in the forecast interval. Thus, if we wish to start at period t and forecast dynamically to $t+n^*$, we would compute a one-step ahead forecast for $t+1$, a two-step ahead forecast for $t+2$, and so forth, up to an n^*-step ahead forecast for $t+n^*$. It may be useful to note that as with n-step ahead forecasting, we simply initialize a Kalman filter at time $t+1$ and run the filter forward additional periods using no additional signal information. For dynamic forecasting, however, only one n-step ahead forecast is required to compute all of the forecast values since the information set is not updated from the beginning of the forecast period.

Smoothed Forecasting

Alternatively, we can compute *smoothed forecasts* which use all available signal data over the forecast sample (for example, $a_{t+n|t+n^*}$). These forward looking forecasts may be computed by initializing the states at the start of the forecast period, and performing a Kalman smooth over the entire forecast period using all relevant signal data. This technique is useful in settings where information on the entire path of the signals is used to interpolate values throughout the forecast sample.

We make one final comment about the forecasting methods described above. For traditional n-step ahead and dynamic forecasting, the states are typically initialized using the one-step ahead forecasts of the states and variances at the start of the forecast window. For smoothed forecasts, one would generally initialize the forecasts using the corresponding smoothed values of states and variances. There may, however, be situations where you wish to choose a different set of initial values for the forecast filter or smoother. The EViews forecasting routines (described in "State Space Procedures" beginning on page 771) provide you with considerable control over these initial settings. Be aware, however, that the interpretation of the forecasts in terms of the available information will change if you choose alternative settings.

Estimation

To implement the Kalman filter and the fixed-interval smoother, we must first replace any unknown elements of the system matrices by their estimates. Under the assumption that the ϵ_t and v_t are Gaussian, the sample log likelihood:

$$\log L(\theta) = -\frac{nT}{2}\log 2\pi - \frac{1}{2}\sum_t \log\left|\tilde{F}_t(\theta)\right| - \frac{1}{2}\sum_t \tilde{\epsilon}_t'(\theta)\tilde{F}_t(\theta)^{-1}\tilde{\epsilon}_t(\theta) \qquad (25.16)$$

may be evaluated using the Kalman filter. Using numeric derivatives, standard iterative techniques may be employed to maximize the likelihood with respect to the unknown parameters θ (see Appendix C, "Estimation and Solution Options", on page 956).

Initial Conditions

Evaluation of the Kalman filter, smoother, and forecasting procedures all require that we provide the initial one-step ahead predicted values for the states $\alpha_{1|0}$ and variance matrix $P_{1|0}$. With some stationary models, steady-state conditions allow us to use the system matrices to solve for the values of $\alpha_{1|0}$ and $P_{1|0}$. In other cases, we may have preliminary estimates of $\alpha_{1|0}$, along with measures of uncertainty about those estimates. But in many cases, we may have no information, or *diffuse priors*, about the initial conditions.

Specifying a State Space Model in EViews

EViews handles a wide range of single and multiple-equation state space models, providing you with detailed control over the specification of your system equations, covariance matrices, and initial conditions.

The first step in specifying and estimating a state space model is to create a state space object. Select **Object/New Object.../Sspace** from the main toolbar or type `sspace` in the command window. EViews will create a state space object and open an empty state space specification window.

There are two ways to specify your state space model. The easiest is to use EViews' special "auto-specification" features to guide you in creating some of the standard forms for these models. Simply press the **AutoSpec** button on the sspace object toolbar. Specialized dialogs will open to guide you through the specification process. We will describe this method in greater detail in "Auto-Specification" on page 766.

The more general method of describing your state space model uses keywords and text to describe the signal equations, state equations, error structure, initial conditions, and if desired, parameter starting values for estimation. The next section describes the general syntax for the state space object.

Specification Syntax

State Equations

A state equation contains the "@STATE" keyword followed by a valid state equation specification. Bear in mind that:

- Each equation must have a unique dependent variable name; expressions are not allowed. Since EViews does not automatically create workfile series for the states, you may use the name of an existing (non-series) EViews object.

- State equations may not contain signal equation dependent variables, or leads or lags of these variables.
- Each state equation must be linear in the one-period lag of the states. Nonlinearities in the states, or the presence of contemporaneous, lead, or multi-period lag states will generate an error message. We emphasize the point that the one-period lag restriction on states *is not restrictive* since higher order lags may be written as new state variables. An example of this technique is provided in the example "ARMAX(2, 3) with a Random Coefficient" on page 762.
- State equations may contain exogenous variables and unknown coefficients, and may be nonlinear in these elements.

In addition, state equations may contain an optional error or error variance specification. If there is no error or error variance, the state equation is assumed to be deterministic. Specification of the error structure of state space models is described in greater detail in "Errors and Variances" on page 760.

Examples

The following two state equations define an unobserved error with an AR(2) process:

```
@state sv1 = c(2)*sv1(-1) + c(3)*sv2(-1) + [var = exp(c(5))]
@state sv2 = sv1(-1)
```

The first equation parameterizes the AR(2) for SV1 in terms of an AR(1) coefficient, C(2), and an AR(2) coefficient, C(3). The error variance specification is given in square brackets. Note that the state equation for SV2 defines the lag of SV1 so that SV2(-1) is the two period lag of SV1.

Similarly, the following are valid state equations:

```
@state sv1 = sv1(-1) + [var = exp(c(3))]
@state sv2 = c(1) + c(2)*sv2(-1) + [var = exp(c(3))]
@state sv3 = c(1) + exp(c(3)*x/z) + c(2)*sv3(-1) + [var =
   exp(c(3))]
```

describing a random walk, and an AR(1) with drift (without/with exogenous variables).

The following are *not* valid state equations:

```
@state exp(sv1) = sv1(-1) + [var = exp(c(3))]
@state sv2 = log(sv2(-1)) + [var = exp(c(3))]
@state sv3 = c(1) + c(2)*sv3(-2) + [var=exp(c(3))]
```

since they violate at least one of the conditions described above (in order: expression for dependent state variable, nonlinear in state, multi-period lag of state variables).

Observation/Signal Equations

By default, if an equation specification is not specifically identified as a state equation using the "@STATE" keyword, it will be treated by EViews as an observation or signal equation. Signal equations may also be identified explicitly by the keyword "@SIGNAL". There are some aspects of signal equation specification to keep in mind:

- Signal equation dependent variables may involve expressions.
- Signal equations may not contain current values or leads of signal variables. You should be aware that any lagged signals are treated as predetermined for purposes of multi-step ahead forecasting (for discussion and alternative specifications, see Harvey 1989, pp. 367-368).
- Signal equations must be linear in the contemporaneous states. Nonlinearities in the states, or the presence of leads or lags of states will generate an error message. Again, the restriction that there are no state lags is not restrictive since additional deterministic states may be created to represent the lagged values of the states.
- Signal equations may have exogenous variables and unknown coefficients, and may be nonlinear in these elements.

Signal equations may also contain an optional error or error variance specification. If there is no error or error variance, the equation is assumed to be deterministic. Specification of the error structure of state space models is described in greater detail in "Errors and Variances" on page 760.

Examples

The following are valid signal equation specifications:

```
log(passenger) = c(1) + c(3)*x + sv1 + c(4)*sv2
@signal y = sv1 + sv2*x1 + sv3*x2 + sv4*y(-1) + [var=exp(c(1))]
z = sv1 + sv2*x1 + sv3*x2 + c(1) + [var=exp(c(2))]
```

The following are invalid equations:

```
log(passenger) = c(1) + c(3)*x + sv1(-1)
@signal y = sv1*sv2*x1 + [var = exp(c(1))]
z = sv1 + sv2*x1 + z(1) + c(1) + [var = exp(c(2))]
```

since they violate at least one of the conditions described above (in order: lag of state variable, nonlinear in a state variable, lead of signal variable).

Errors and Variances

While EViews always adds an implicit error term to each equation in an equation or system object, the handling of error terms differs in a sspace object. In a sspace object, the

equation specifications in a signal or state equation do not contain error terms unless specified explicitly.

The easiest way to add an error to a state space equation is to specify an implied error term using its variance. You can simply add an error variance expression, consisting of the keyword "VAR" followed by an assignment statement (all enclosed in square brackets), to the existing equation:

```
@signal y = c(1) + sv1 + sv2 + [var = 1]
@state sv1 = sv1(-1) + [var = exp(c(2))]
@state sv2 = c(3) + c(4)*sv2(-1) + [var = exp(c(2)*x)]
```

The specified variance may be a known constant value, or it can be an expression containing unknown parameters to be estimated. You may also build time-variation into the variances using a series expression. Variance expressions may not, however, contain state or signal variables.

While straightforward, this direct variance specification method does not admit correlation between errors in different equations (by default, EViews assumes that the covariance between error terms is 0). If you require a more flexible variance structure, you will need to use the "named error" approach to define named errors with variances and covariances, and then to use these named errors as parts of expressions in the signal and state equations.

The first step of this general approach is to define your named errors. You may declare a named error by including a line with the keyword "@ENAME" followed by the name of the error:

```
@ename e1
@ename e2
```

Once declared, a named error may enter linearly into state and signal equations. In this manner, one can build correlation between the equation errors. For example, the errors in the state and signal equations in the sspace specification:

```
y = c(1) + sv1*x1 + e1
@state sv1 = sv1(-1) + e2 + c(2)*e1
@ename e1
@ename e2
```

are, in general, correlated since the named error E1 appears in both equations.

In the special case where a named error is the only error in a given equation, you can both declare and use the named residual by adding an error expression consisting of the keyword "ENAME" followed by an assignment and a name identifier:

```
y = c(1) + sv1*x1 + [ename = e1]
@state sv1 = sv1(-1) + [ename = e2]
```

The final step in building a general error structure is to define the variances and covariances associated with your named errors. You should include a sspace line comprised of the keyword "@EVAR" followed by an assignment statement for the variance of the error or the covariance between two errors:

```
@evar cov(e1, e2) = c(2)
@evar var(e1) = exp(c(3))
@evar var(e2) = exp(c(4))*x
```

The syntax for the @EVAR assignment statements should be self-explanatory. Simply indicate whether the term is a variance or covariance, identify the error(s), and enter the specification for the variance or covariance. There should be a separate line for each named error covariance or variance that you wish to specify. If an error term is named, but there are no corresponding "VAR=" or @EVAR specifications, the missing variance or covariance specifications will remain at the default values of "NA" and "0", respectively.

As you might expect, in the special case where an equation contains a single error term, you may combine the named error and direct variance assignment statements:

```
@state sv1 = sv1(-1) + [ename = e1, var = exp(c(3))]
@state sv2 = sv2(-1) + [ename = e2, var = exp(c(4))]
@evar cov(e1, e2) = c(5)
```

Specification Examples

ARMAX(2, 3) with a Random Coefficient

We can use the syntax described above to define an ARMAX(2,3) with a random coefficient for the regression variable X:

```
y = c(1) + sv5*x + sv1 + c(4)*sv2 + c(5)*sv3 + c(6)*sv4
@state sv1 = c(2)*sv1(-1) + c(3)*sv2(-1) + [var=exp(c(7))]
@state sv2 = sv1(-1)
@state sv3 = sv2(-1)
@state sv4 = sv3(-1)
@state sv5 = sv5(-1) + [var=3]
```

The AR coefficients are parameterized in terms of C(2) and C(3), while the MA coefficients are given by C(4), C(5) and C(6). The variance of the innovation is restricted to be a positive function of C(7). SV5 is the random coefficient on X, with variance restricted to be 3.

Recursive and Random Coefficients

The following example describes a model with one random coefficient (SV1), one recursive coefficient (SV2), and possible correlation between the errors for SV1 and Y:

```
y = c(1) + sv1*x1 + sv2*x2 + [ename = e1, var = exp(c(2))]
@state sv1 = sv1(-1) + [ename = e2, var = exp(c(3)*x)]
@state sv2 = sv2(-1)
@evar cov(e1,e2) = c(4)
```

The variances and covariances in the model are parameterized in terms of the coefficients C(2), C(3) and C(4), with the variances of the observed Y and the unobserved state SV1 restricted to be non-negative functions of the parameters.

Parameter Starting Values

Unless otherwise instructed, EViews will initialize all parameters to the current values in the corresponding coefficient vector or vectors. As in the system object, you may override this default behavior by specifying explicitly the desired values of the parameters using a PARAM or @PARAM statement. For additional details, see "Starting Values" on page 703.

Specifying Initial Conditions

By default, EViews will handle the initial conditions for you. For some stationary models, steady-state conditions allow us to solve for the values of α_0 and P_0. For cases where it is not possible to solve for the initial conditions, EViews will treat the initial values as diffuse, setting $\alpha_{1|0} = 0$, and $P_{1|0}$ to an arbitrarily high number to reflect our uncertainty about the values (see "Technical Discussion" on page 775).

You may, however have prior information about the values of $\alpha_{1|0}$ and $P_{1|0}$. In this case, you can create a vector or matrix that contains the appropriate values, and use the "@MPRIOR" or "@VPRIOR" keywords to perform the assignment.

To set the initial states, enter "@MPRIOR" followed by the name of a vector object. The length of the vector object must match the state dimension. The order of elements should follow the order in which the states were introduced in the specification screen.

```
@mprior v1
@vprior m1
```

To set the initial state variance matrix, enter "@VPRIOR" followed by the name of a sym object (note that it must be a sym object, and not an ordinary matrix object). The dimensions of the sym must match the state dimension, with the ordering following the order in which the states appear in the specification. If you wish to set a specific element to be diffuse, simply assign the element the "NA" missing value. EViews will reset all of the corresponding variances and covariances to be diffuse.

For example, suppose you have a two equation state space object named SS1 and you want to set the initial values of the state vector and the state variance matrix as:

$$\begin{bmatrix} SV1 \\ SV2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad \text{var}\begin{bmatrix} SV1 \\ SV2 \end{bmatrix} = \begin{bmatrix} 1 & 0.5 \\ 0.5 & 2 \end{bmatrix} \tag{25.17}$$

First, create a named vector object, say SVEC0, to hold the initial values. Click **Object/New Object**, choose **Matrix-Vector-Coef** and enter the name SVEC0. Click **OK**, and then choose the type **Vector** and specify the size of the vector (in this case 2 rows). When you click **OK**, EViews will display the spreadsheet view of the vector SVEC0. Click the **Edit +/–** button to toggle on edit mode and type in the desired values. Then create a named matrix object, say SVAR0, in an analogous fashion.

Alternatively, you may find it easier to create and initialize the vector and matrix using commands. You can enter the following commands in the command window:

```
vector(2) svec0
svec0.fill 1, 0
matrix(2,2) svar0
svar0.fill(b=c) 1, 0.5, 0.5, 2
```

Then, simply add the lines:

```
@mprior svec0
@vprior svar0
```

to your sspace object by editing the specification window. Alternatively, you can type the following commands in the command window:

```
ss1.append @mprior svec0
ss1.append @vprior svar0
```

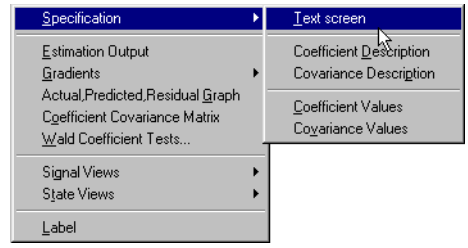
For more details on matrix objects and the `fill` and `append` commands, see Chapter 3, "Matrix Language", on page 23 of the *Command and Programming Reference*.

Specification Views

State space models may be very complex. To aid you in examining your specification, EViews provides views which allow you to view the text specification in a more compact form, and to examine the numerical values of your system matrices evaluated at current parameter values.

Click on the **View** menu and select **Specification...** The following Specification views are always available, regardless of whether the sspace has previously been estimated:

- **Text Screen**. This is the familiar text view of the specification. You should use this view when you create or edit the state space specification. This view may also be accessed by clicking on the **Spec** button on the sspace toolbar.



- **Coefficient Description**. Text description of the structure of your state space specification. The variables on the left-hand side, representing α_{t+1} and y_t, are expressed as linear functions of the state variables α_t, and a remainder term CONST. The elements of the matrix are the corresponding coefficients. For example, the ARMAX example has the following Coefficient Description view:

Sspace: SS_ARMAX23 Workfile: AIRLINE\Airline

View | Proc | Object | Print | Name | Freeze | Spec | Estimate | Stats | Forecast

Structural Coefficient Description

	CONST	SV1	SV2	SV3	SV4	SV5
SV1(1)	0	C(2)	C(3)	0	0	0
SV2(1)	0	1	0	0	0	0
SV3(1)	0	0	1	0	0	0
SV4(1)	0	0	0	1	0	0
SV5(1)	0	0	0	0	0	1
Y	C(1)	1	C(4)	C(5)	C(6)	X

- **Covariance Description**. Text description of the covariance matrix of the state space specification. For example, the ARMAX example has the following Covariance Description view:

Sspace: SS_ARMAX23 Workfile: AIRLINE\Airline

View | Proc | Object | Print | Name | Freeze | Spec | Estimate | Stats | Forecast

Structural Covariance Description

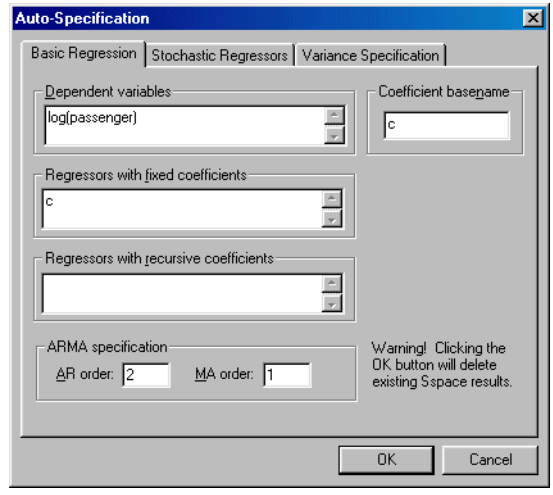
	SV1	SV2	SV3	SV4	SV5	Y
SV1	EXP(C(7))	0	0	0	0	0
SV2	0	0	0	0	0	0
SV3	0	0	0	0	0	0
SV4	0	0	0	0	0	0
SV5	0	0	0	0	3	0
Y	0	0	0	0	0	0

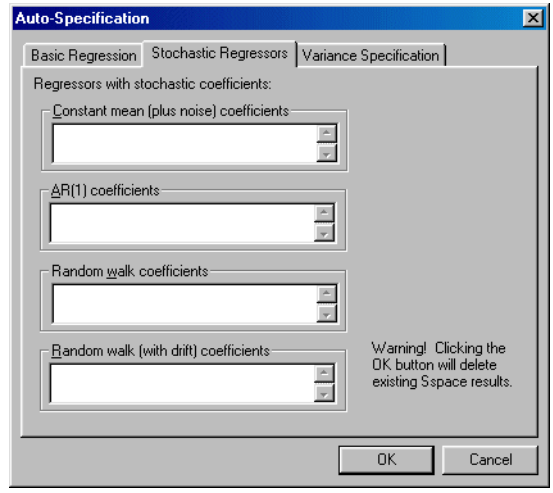
- **Coefficient Values**. Numeric description of the structure of the signal and the state equations evaluated at current parameter values. If the system coefficient matrix is time-varying, EViews will prompt you for a date/observation at which to evaluate the matrix.
- **Covariance Values**. Numeric description of the structure of the state space specification evaluated at current parameter values. If the system covariance matrix is time-varying, EViews will prompt you for a date/observation at which to evaluate the matrix.

Auto-Specification

To aid you in creating a state space specification, EViews provides you with "auto-specification" tools which will create the text representation of a model that you specify using dialogs. This tool may be very useful if your model is a standard regression with fixed, recursive, and various random coefficient specifications, and/or your errors have a general ARMA structure.

Click on the **AutoSpec** button on the sspace toolbar, or select **Proc/Define State Space...** from the menu. EViews opens a three tab dialog. The first tab is used to describe the basic regression portion of your specification. Enter the dependent variable, and any regressors which have fixed or recursive coefficients. You can choose which COEF object EViews uses for indicating unknowns when setting up the specification. At the bottom, you can specify an ARMA structure for your errors. Here, we have specified a simple ARMA(2,1) specification for LOG(PASSENGER).





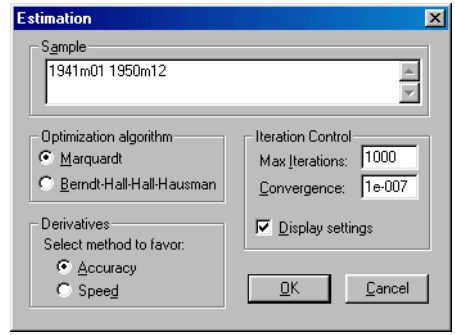
The second tab of the dialog is used to add any regressors which have random coefficients. Simply enter the appropriate regressors in each of the four edit fields. EViews allows you to define regressors with any combination of constant mean, AR(1), random walk, or random walk (with drift) coefficients.

Lastly, the **Auto-Specification** dialog allows you to choose between basic variance structures for your state space model. Click on the **Variance Specification** tab, and choose between an identity matrix, common diagonal (diagonal with common variances), diagonal, or general (unrestricted) variance matrix for the signals and for the states. The dialog also allows you to allow the signal equation(s) and state equations(s) to have non-zero error covariances.

We emphasize the fact that your sspace object is not restricted to the choices provided in this dialog. If you find that the set of specifications supported by **Auto-Specification** is too restrictive, you may use it the dialogs as a tool to build a basic specification, and then edit the specification to describe your model.

Estimating a State Space Model

Once you have specified a state space model and verified that your specification is correct, you are ready to estimate the model. To open the estimation dialog, simply click on the **Estimate** button on the toolbar or select **Proc/Estimate...**



As with other estimation objects, EViews allows you to set the estimation sample, the maximum number of iterations, convergence tolerance, the estimation algorithm, derivative settings and whether to display the starting values. The default settings should provide a good start for most problems; if you choose to change the settings, see "Setting Estimation Options" on page 951 for related discussion of estimation options. When you click on **OK**, EViews will begin estimation using the specified settings.

There are two additional things to keep in mind when estimating your model:

- Although the EViews Kalman filter routines will automatically handle any missing values in your sample, EViews does require that your estimation sample be contiguous, with no gaps between successive observations.
- If there are no unknown coefficients in your specification, you will still have to "estimate" your sspace to run the Kalman filter and initialize elements that EViews needs in order to perform further analysis.

Interpreting the estimation results

After you choose the variance options and click **OK**, EViews presents the estimation results in the state space window. For example, if we specify an ARMA(2,1) for the log of the monthly international airline passenger totals from January 1949 to December 1960 (from Box and Jenkins, 1976, series G, p. 531):

```
log(passenger) = c(1) + sv1 + c(4)*sv2
@state sv1 = c(2)*sv1(-1) + c(3)*sv2(-1) + [var=exp(c(5))]
@state sv2 = sv1(-1)
```

and estimate the model, EViews will open the estimation output view:

Sspace: SS_ARMA21
Estimation Method: Maximum Likelihood (Marquardt)
Date: 11/12/99 Time: 11:58
Sample: 1949M01 1960M12
Included observations: 144
Convergence achieved after 55 iterations

	Coefficient	Std. Error	z-Statistic	Prob.
C(1)	5.499767	0.257517	21.35687	0.0000
C(2)	0.409013	0.167201	2.446239	0.0144
C(3)	0.547165	0.164608	3.324055	0.0009
C(4)	0.841481	0.100167	8.400800	0.0000
C(5)	-4.589401	0.172696	-26.57501	0.0000

	Final State	Root MSE	z-Statistic	Prob.
SV1	0.267125	0.100792	2.650274	0.0080
SV2	0.425488	0.000000	NA	1.0000

Log likelihood	124.3366	Parameters	5
Akaike info criterion	-1.629674	Likelihood observations	144
Schwarz criterion	-1.485308	Missing observations	0
Hannan-Quinn criter.	-1.571012	Partial observations	0
		Diffuse priors	0

The bulk of the output view should be familiar from other EViews estimation objects. The information at the top describes the basics of the estimation: the name of the sspace object, estimation method, the date and time of estimation, sample and number of objects in the sample, convergence information, and the coefficient estimates. The bottom part of

the view reports the maximized log likelihood value, the number of estimated parameters, and the associated information criteria.

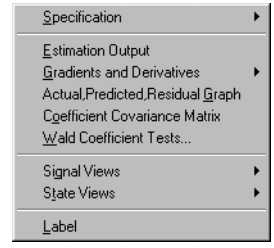
Some parts of the output, however, are new and may require discussion. The bottom section provides additional information about the handling of missing values in estimation. "Likelihood observations" reports the actual number of observations that are used in forming the likelihood. This number (which is the one used in computing the information criteria) will differ from the "Included observations" reported at the top of the view when EViews drops an observation from the likelihood calculation because all of the signal equations have missing values. The number of omitted observations is reported in "Missing observations". "Partial observations" reports the number of observations that are included in the likelihood, but for which some equations have been dropped. "Diffuse priors" indicates the number of initial state covariances for which EViews is unable to solve and for which there is no user initialization. EViews' handling of initial states and covariances is described in greater detail in "Initial Conditions" on page 775.

EViews also displays the final one-step ahead values of the state vector, $\alpha_{T+1|T}$, and the corresponding RMSE values (square roots of the diagonal elements of $P_{T+1|T}$). For settings where you may care about the entire path of the state vector and covariance matrix, EViews provides you with a variety of views and procedures for examining the state results in greater detail.

Working with the State Space

EViews provides a variety of specialized tools for specifying and examining your state space specification. As with other estimation objects, the sspace object provides additional views and procedures for examining the estimation results, performing inference and specification testing, and extracting results into other EViews objects.

State Space Views



Many of the state space views should be familiar from previous discussion:

- We have already discussed the **Specification...** views in our analysis of "Specification Views" on page 764.
- The **Estimation Output** view displays the coefficient estimates and summary statistics as described above in "Interpreting the estimation results" on page 768. You may also access this view by pressing **Stats** on the sspace toolbar.
- The **Gradients and Derivatives...** views should be familiar from other estimation objects. If the sspace contains parameters to be estimated, this view provides sum-

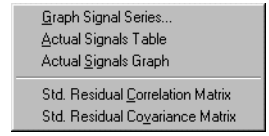
mary and visual information about the gradients of the log likelihood at estimated parameters (if the sspace is estimated) or at current parameter values.

- **Actual, Predicted, Residual Graph** displays, in graphical form, the actual and one-step ahead fitted values of the signal dependent variable(s), $y_{t|t-1}$, and the one-step ahead standardized residuals, $e_{t|t-1}$.
- Select **Coefficient Covariance Matrix** to view the estimated coefficient covariance.
- **Wald Coefficient Tests...** allows you to perform hypothesis tests on the estimated coefficients. For details, see "Wald Test (Coefficient Restrictions)" on page 572.
- **Label** allows you to annotate your object. See "Labeling Objects" on page 82.

Note that with the exception of the **Label** and **Specification...** views, these views are available only following successful estimation of your state space model.

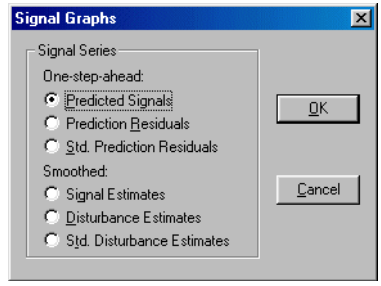
Signal Views

When you click on **View/Signal Views**, EViews displays a submenu containing additional view selections. Two of these selections are always available, even if the state space model has not yet been estimated:



- **Actual Signal Table** and **Actual Signal Graph** display the dependent signal variables in spreadsheet and graphical forms, respectively. If there are multiple signal equations, EViews will display a each series with its own axes.

  The remaining views are only available following estimation.



- **Graph Signal Series...** opens a dialog with choices for the results to be displayed. The dialog allows you to choose between the one-step ahead predicted signals, $y_{t|t-1}$, the corresponding one-step residuals, $\epsilon_{t|t-1}$, or standardized one-step residuals, $e_{t|t-1}$, the smoothed signals, $\hat{y}_t$, smoothed signal disturbances, $\hat{\epsilon}_t$, or the standardized smoothed signal disturbances, $\hat{e}_t$. ± 2 (root mean square) standard error bands are plotted where appropriate.
- **Std. Residual Correlation Matrix** and **Std. Residual Covariance Matrix** display the correlation and covariance matrix of the standardized one-step ahead signal residual, $e_{t|t-1}$.

State Views

To examine the unobserved state components, click on **View/ State Views** to display the state submenu. EViews allows you to examine the initial or final values of the state components, or to graph the full time-path of various filtered or smoothed state data.

Two of the views are available either before or after estimation:

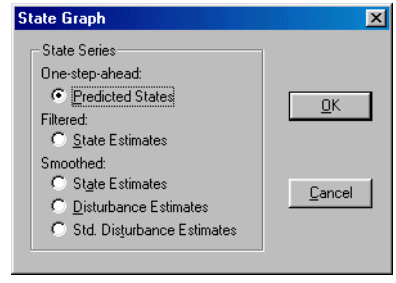
- **Initial State Vector** and **Initial State Covariance Matrix** display the values of the initial state vector, α_0, and covariance matrix, P_0. If the unknown parameters have previously been estimated, EViews will evaluate the initial conditions using the estimated values. If the sspace has not been estimated, the current coefficient values will be used in evaluating the initial conditions.

  This information is especially relevant in models where EViews is using the current values of the system matrices to solve for the initial conditions. In cases where you are having difficulty starting your estimation, you may wish to examine the values of the initial conditions at the starting parameter values for any sign of problems.

The remainder of the views are only available following successful estimation:

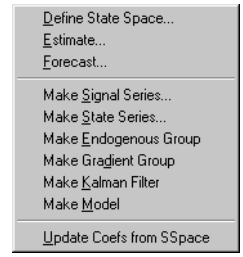
- **Final State Vector** and **Final State Covariance Matrix** display the values of the final state vector, α_T, and covariance matrix, P_T, evaluated at the estimated parameters.

- Select **Graph State Series...** to display a dialog containing several choices for the state information. You can graph the one-step ahead predicted states, $a_{t|t-1}$, the filtered (contemporaneous) states, a_t, the smoothed state estimates, $\hat{\alpha}_t$, smoothed state disturbance estimates, $\hat{v}_t$, or the standardized smoothed state disturbances, $\hat{\eta}_t$. In each case, the data are displayed along with corresponding ± 2 standard error bands.



State Space Procedures

You can use the EViews procedures to create, estimate, forecast, and generate data from your state space specification. Select **Proc** in the sspace toolbar to display the available procedures:



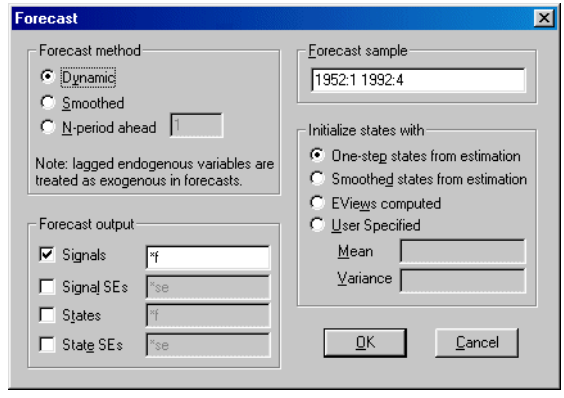
- **Define State Space...** calls up the **Auto-Specification** dialog (see "Auto-Specification" on page 766). This feature provides a method of specifying a variety of common state space specifications using interactive menus.
- Select **Estimate...** to estimate the parameters of the specification (see "Estimating a State Space Model" on page 767).

These above items are available both before and after estimation. The automatic specification tool will replace the existing state space specification and will clear any results.

Once you have estimated your sspace, EViews provides additional tools for generating data:

- The **Forecast...** dialog allows you to generate forecasts of the states, signals, and the associated standard errors using alternative methods and initialization approaches.

  First, select the forecast method. You can select between dynamic, smoothed, and n-period ahead forecasting, as described in "Forecasting" on page 756. Note that any lagged endogenous variables on the right-hand side of your signal equations will be treated as predetermined for purposes of forecasting.

  EViews allows you to save various types of forecast output in series in your workfile. Simply check any of the output boxes, and specify the names for the series in the corresponding edit field.

  You may specify the names either as a list or using a wildcard expression. If you choose to list the names, the number of identifiers must match the number of signals in your specification. You should be aware that if an output series with a specified name already exists in the workfile, EViews will overwrite the entire contents of the series.

  If you use a wildcard expression, EViews will substitute the name of each signal in the appropriate position in the wildcard expression. For example, if you have a model with signals Y1 and Y2, and elect to save the one-step predictions in "PRED*", EViews will use the series PREDY1 and PREDY2 for output. There are two limitations to this feature: (1) you may not use the wildcard expression "*" to save

signal results since this will overwrite the original signal data, and (2) you may not use a wildcard when any signal dependent variables are specified by expression, or when there are multiple equations for a signal variable. In both cases, EViews will be unable to create the new series and will generate an error message.

Keep in mind that if your signal dependent variable is an expression, EViews will only provide forecasts of the expression. Thus, if your signal variable is LOG(Y), EViews will forecast the logarithm of Y.

Now enter a sample and specify the treatment of the initial states, and then click **OK**. EViews will compute the forecast and will place the results in the specified series. No output window will open.

There are several options available for setting the initial conditions. If you wish, you can instruct the sspace object to use the **One-step ahead** or **Smoothed** estimates of the state and state covariance as initial values for the forecast period. The two initialization methods differ in the amount of information used from the estimation sample; one-step ahead uses information up to the beginning of the forecast period, while smoothed uses the entire estimation period.

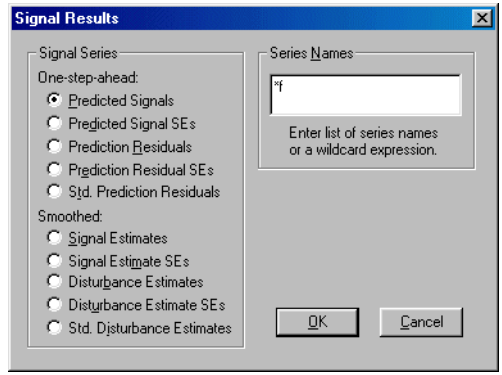
Alternatively, you may use **EViews computed** initial conditions. As in estimation, if possible, EViews will solve the Algebraic Riccati equations to obtain values for the initial state and state covariance at the start of each forecast interval. If solution of these conditions is not possible, EViews will use diffuse priors for the initial values.

Lastly, you may choose to provide a vector and sym object which contain the values for the forecast initialization. Simply select **User Specified** and enter the name of valid EViews objects in the appropriate edit fields.

Note that when performing either dynamic or smoothed forecasting, EViews requires that one-step ahead and smoothed initial conditions be computed from the estimation sample. If you choose one of these two forecasting methods and your forecast period begins either before or after the estimation sample, EViews will issue an error and instruct you to select a different initialization method.

When computing n-step ahead forecasting, EViews will adjust the start of the forecast period so that it is possible to obtain initial conditions for each period using the specified method. For the one-step ahead and smoothed methods, this means that at the earliest, the forecast period will begin $n - 1$ observations into the estimation sample, with earlier forecasted values set to NA. For the other initialization methods, forecast sample endpoint adjustment is not required.

- **Make Signal Series...** allows you to create series containing various signal results computed over the estimation sample. Simply click on the menu entry to display the results dialog.

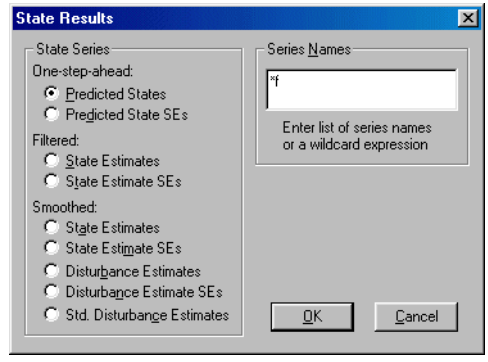


You may select the one-step ahead predicted signals, $\tilde{y}y_{t|t-1}$, one-step prediction residuals, $\epsilon_{t|t-1}$, smoothed signal, $\hat{y}_t$, or signal disturbance estimates, $\hat{\epsilon}_t$. EViews also allows you to save the corresponding standard errors for each of these components (square roots of the diagonal elements of $F_{t|t-1}$, S_t, and $\hat{\Omega}_t$), or the standardized values of the one-step residuals and smoothed disturbances, $e_{t|t-1}$ or $\hat{e}_t$.

Next, specify the names of your series in the edit field using a list or wildcards as described above. Click **OK** to generate a group containing the desired signal series.

As above, if your signal dependent variable is an expression, EViews will only export results based upon the entire expression.

- **Make State Series...** opens a dialog allowing you to create series containing results for the state variables computed over the estimation sample. You can choose to save either the one-step ahead state estimate, $a_{t|t-1}$, the filtered state mean, a_t, the smoothed states, $\hat{\alpha}_t$, state disturbances, $\hat{v}_t$, standardized state disturbances, $\hat{\eta}_t$, or the corresponding standard error series (square roots of the diagonal elements of $P_{t|t-1}$, P_t, V_t and $\hat{\Omega}_t$).



Simply select one of the output types, and enter the names of the output series in the edit field. The rules for specifying the output names are the same as for the **Forecast...** procedure described above. Note that the wildcard expression "*" is permitted when saving state results. EViews will simply use the state names defined in your specification.

We again caution you that if an output series exists in the workfile, EViews will overwrite the entire contents of the series.

- Click on **Make Endogenous Group** to create a group object containing the signal dependent variable series.

- **Make Gradient Group** creates a group object with series containing the gradients of the log likelihood. These series are named "GRAD##" where ## is a unique number in the workfile.
- **Make Kalman Filter** creates a new state space object containing the current specification, but with all parameters replaced by their estimated values. In this way you can "freeze" the current state space for additional analysis. This procedure is similar to the **Make Model** procedure found in other estimation objects.
- **Make Model** creates a model object containing the state space equations.
- **Update Coefs from Sspace** will place the estimated parameters in the appropriate coefficient vectors.

Converting from Version 3 Sspace

Those of you who have worked with the EViews Version 3 sspace object will undoubtedly be struck by the large number of changes and additional features in Version 4 and later. In addition to new estimation options, views and procedures, we have changed the underlying specification syntax to provide you with considerable additional flexibility. A wide variety of specifications that were not supported in earlier versions may be estimated with the current sspace object.

The cost of these additional features and added flexibility is that Version 3 sspace objects are not fully compatible with those in the current version. This has two important practical effects:

- If you load in a workfile which contains a Version 3 sspace object, *all previous estimation results will be cleared* and the text of the specification will be translated to the current syntax. The original text will be retained as comments at the bottom of your sspace specification.
- If you take a workfile which contains a new sspace object created with EViews 4 or later and attempt to read it into an earlier version of EViews, the object will not be read, and EViews will warn you that a partial load of the workfile was performed. If you subsequently save the workfile, *the original sspace object will not be saved with the workfile*.

Technical Discussion

Initial Conditions

If there are no @MPRIOR or @VPRIOR statements in the specification, EViews will either: (1) solve for the initial state mean and variance, or (2) initialize the states and variances using diffuse priors.

Solving for the initial conditions is only possible if the state transition matrices T, and variance matrices P and Q are non time-varying and satisfy certain stability conditions (see Harvey, p. 121). If possible, EViews will solve for the conditions $P_{1|0}$ using the familiar relationship: $(I - T \otimes T) \times \text{vec}(P) = \text{vec}(Q)$. If this is not possible, the states will be treated as diffuse unless otherwise specified.

When using diffuse priors, EViews follows the method adopted by Koopman, Shephard and Doornik (1999) in setting $\alpha_{1|0} = 0$, and $P_{1|0} = \kappa I_M$, where the κ is an arbitrarily chosen large number. EViews uses the authors' recommendation that one first set $\kappa = 10^6$ and then adjust it for scale by multiplying by the largest diagonal element of the residual covariances.

Chapter 26. Models

A model in EViews is a set of one or more equations that jointly describe the relationship between a set of variables. The model equations can come from many sources: they can be simple identities, they can be the result of estimation of single equations, or they can be the result of estimation using any one of EViews' multiple equation estimators.

EViews models allow you to combine equations from all these sources inside a single object, which may be used to create a deterministic or stochastic joint forecast or simulation for all of the variables in the model. In a deterministic setting, the inputs to the model are fixed at known values, and a single path is calculated for the output variables. In a stochastic environment, uncertainty is incorporated into the model by adding a random element to the coefficients, the equation residuals or the exogenous variables.

Models also allow you to examine simulation results under different assumptions concerning the variables that are determined outside the model. In EViews, we refer to these sets of assumptions as *scenarios*, and provide a variety of tools for working with multiple model scenarios.

Even if you are working with only a single equation, you may find that it is worth creating a model from that equation so that you may use the features provided by the EViews Model object.

Overview

The following section provides a brief introduction to the purpose and structure of the EViews model object, and introduces terminology that will be used throughout the rest of the chapter.

A model consists of a set of *equations* that describe the relationships between a set of *variables*.

The variables in a model can be divided into two categories: those determined inside the model, which we refer to as the *endogenous variables*, and those determined outside the model, which we refer to as the *exogenous variables*. A third category of variables, the *add factors*, are a special case of exogenous variables.

In its most general form, a model can be written in mathematical notation as:

$$F(y, x) = 0 \tag{26.1}$$

where y is the vector of endogenous variables, x is the vector of exogenous variables, and F is a vector of real-valued functions $f_i(y, x)$. For the model to have a unique solution, there should typically be as many equations as there are endogenous variables.

In EViews, each equation in the model must have a unique endogenous variable assigned to it. That is, each equation in the model must be able to be written in the form:

$$y_i = f_i(y, x) \tag{26.2}$$

where y_i is the endogenous variable assigned to equation i. EViews has the ability to *normalize* equations involving simple transformations of the endogenous variable, rewriting them automatically into explicit form when necessary. Any variable that is not assigned as the endogenous variable for any equation is considered exogenous to the model.

Equations in an EViews model can either be *inline* or *linked*. An inline equation contains the specification for the equation as text within the model. A linked equation is one that brings its specification into the model from an external EViews object such as a single or multiple equation estimation object, or even another model. Linking allows you to couple a model more closely with the estimation procedure underlying the equations, or with another model on which it depends. For example, a model for industry supply and demand might link to another model and to estimated equations:

Industry Supply And Demand Model
← link to macro model object for forecasts of total consumption
← link to equation object containing industry supply equation
← link to equation object containing industry demand equation
← inline identity: supply = demand

Equations can also be divided into *stochastic equations* and *identities*. Roughly speaking, an identity is an equation that we would expect to hold exactly when applied to real world data, while a stochastic equation is one that we would expect to hold only with random error. Stochastic equations typically result from statistical estimation procedures while identities are drawn from accounting relationships between the variables.

The most important operation performed on a model is to *solve* the model. By solving the model, we mean that for a given set of values of the exogenous variables, X, we will try to find a set of values for the endogenous variables, Y, so that the equations in the model are satisfied within some numerical tolerance. Often, we will be interested in solving the model over a sequence of periods, in which case, for a simple model, we will iterate through the periods one by one. If the equations of the model contain future endogenous

variables, we may require a more complicated procedure to solve for the entire set of periods simultaneously.

In EViews, when solving a model, we must first associate data with each variable in the model by *binding* each of the model variables to a series in the workfile. We then solve the model for each observation in the selected sample and place the results in the corresponding series.

When binding the variables of the model to specific series in the workfile, EViews will often modify the name of the variable to generate the name of the series. Typically, this will involve adding an extension of a few characters to the end of the name. For example, an endogenous variable in the model may be called "Y", but when EViews solves the model, it may assign the result into an observation of a series in the workfile called "Y_0". We refer to this mapping of names as *aliasing*. Aliasing is an important feature of an EViews model, as it allows the variables in the model to be mapped into different sets of workfile series, without having to alter the equations of the model.

When a model is solved, aliasing is typically applied to the endogenous variables so that historical data is not overwritten. Furthermore, for models which contain lagged endogenous variables, aliasing allows us to bind the lagged variables to either the actual historical data, which we refer to as a *static forecast*, or to the values solved for in previous periods, which we refer to as a *dynamic forecast*. In both cases, the lagged endogenous variables are effectively treated as exogenous variables in the model when solving the model for a single period.

Aliasing is also frequently applied to exogenous variables when using *model scenarios*. Model scenarios allow you to investigate how the predictions of your model vary under different assumptions concerning the path of exogenous variables or add factors. In a scenario, you can change the path of an exogenous variable by overriding the variable. When a variable is o*verridden*, the values for that variable will be fetched from a workfile series specific to that scenario. The name of the series is formed by adding a suffix associated with the scenario to the variable name. This same suffix is also used when storing the solutions of the model for the scenario. By using scenarios it is easy to compare the outcomes predicted by your model under a variety of different assumptions without having to edit the structure of your model.

The following table gives a typical example of how model aliasing might map variable names in a model into series names in the workfile:

Model Variable		Workfile Series	
endogenous Y	→	Y	historical data
	→	Y_0	baseline solution
	→	Y_1	scenario 1
exogenous X	→	X	historical data followed by baseline forecast
	→	X_1	overridden forecast for scenario 1

Earlier, we mentioned a third category of variables called *add factors*. An add factor is a special type of exogenous variable that is used to shift the results of a stochastic equation to provide a better fit to historical data or to fine-tune the forecasting results of the model. While there is nothing that you can do with an add factor that could not be done using exogenous variables, EViews provides a separate interface for add factors to facilitate a number of common tasks.

An Example Model

In this section, we demonstrate how we can use the EViews model object to implement a simple macroeconomic model of the U.S. economy. The specification of the model is taken from Pindyck and Rubinfeld (1998, p. 390). We have provided the data and other objects relating to the model in the sample workfile MACROMOD.WF1. You may find it useful to follow along with the steps in the example, and you can use the workfile to experiment further with the model object.

The macro model contains three stochastic equations and one identity. In EViews notation, these can be written:

```
cn = c(1) + c(2)*y + c(3)*cn(-1)
i = c(4) + c(5)*(y(-1)-y(-2)) + c(6)*y + c(7)*r(-4)
r = c(8) + c(9)*y + c(10)*(y-y(-1)) + c(11)*(m-m(-1)) + c(12)*
    (r(-1)+r(-2))
y = cn + i + g
```

where:

- CN is real personal consumption
- I is real private investment
- G is real government expenditure
- Y is real GDP less net exports

- R is the interest rate on three-month treasury bills
- M is the real money supply, narrowly defined (M1)

and the C(i) are the unknown coefficients.

The model follows the structure of a simple textbook ISLM macroeconomic model, with expenditure equations relating consumption and investment to GDP and interest rates, and a money market equation relating interest rates to GDP and the money supply. The fourth equation is the national accounts expenditure identity which ensures that the components of GDP add to total GDP. The model differs from a typical textbook model in its more dynamic structure, with many of the variables appearing in lagged or differenced form.

To begin, we must first estimate the unknown coefficients in the stochastic equations. For simplicity, we estimate the coefficients by simple single equation OLS. Note that this approach is not strictly valid, since Y appears on the right-hand side of several of the equations as an independent variable but is endogenous to the system as a whole. Because of this, we would expect Y to be correlated with the residuals of the equations, which violates the assumptions of OLS estimation. To adjust for this, we would need to use some form of instrumental variables or system estimation (for details, see the discussion of single equation "Two-stage Least Squares" beginning on page 473 and system "Two-Stage Least Squares" and related sections beginning on page 697).

To estimate the equations in EViews, we create three new equation objects in the workfile (using **Object/New Object.../Equation**), and then enter the appropriate specifications. Since all three equations are linear, we can specify them using list form. To minimize confusion, we will name the three equations according to their endogenous variables. The resulting names and specifications are:

Equation EQCN:	cn c y cn(-1)
Equation EQI:	i c y(-1)-y(-2) y r(-4)
Equation EQR:	r c y y-y(-1) m-m(-1) r(-1) + r(-2)

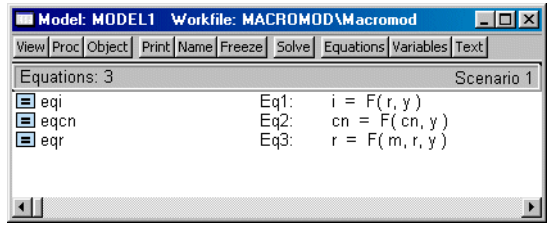
The three equations estimate satisfactorily and provide a reasonably close fit to the data, although much of the fit probably comes from the lagged endogenous variables. The consumption and investment equations show signs of heteroskedasticity, possibly indicating that we should be modeling the relationships in log form. All three equations show signs of serial correlation. We will ignore these problems for the purpose of this example, although you may like to experiment with alternative specifications and compare their performance.

Now that we have estimated the three equations, we can proceed to the model itself. To create the model, we simply select **Object/New Object.../Model** from the menus. To keep the model permanently in the workfile, we name the model by clicking on the **Name** button, enter the name MODEL1, and click on **OK**.

When first created, the model object defaults to *equation view*. Equation view allows us to browse through the specifications and properties of the equations contained in the model. Since we have not yet added any equations to the model, this window will appear empty.

To add our estimated stochastic equations to the model, we can simply copy-and-paste them across from the workfile window. To copy-and-paste, first select the objects in the workfile window, and then use **Edit/Copy** or the right mouse button menu to copy the objects to the clipboard. Click anywhere in the model object window, and use **Edit/Paste** or the right mouse button menu to paste the objects into the model object window.

Alternatively, we could have combined the two steps by first highlighting the three equations, right-mouse clicking, and selecting **Open as Model**. EViews will create a new unnamed model containing the three equations. Press on the **Name** button to name the model object.


Model: MODEL1 Workfile: MACROMOD\Macromod
View Proc Object Print Name Freeze Solve Equations Variables Text
Equations: 3
Scenario 1
eqi
eqcn
eqr
Eq1: i = F(r, y)
Eq2: cn = F(cn, y)
Eq3: r = F(m, r, y)

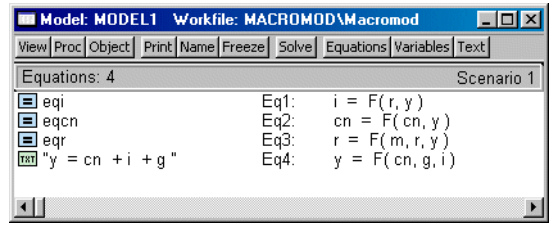
The three estimated equations should now appear in the equation window. Each equation appears on a line with an icon showing the type of object, its name, its equation number, and a symbolic representation of the equation in terms of the variables that it contains. Double clicking on any equation will bring up a dialog of properties of that equation. For the moment, we do not need to alter any of these properties.

We have added our three equations as linked equations. This means if we go back and reestimate one or more of the equations, we can automatically update the equations in the model to the new estimates by using the procedure **Proc/Links/Update All Links**.

To complete the model, we must add our final equation, the national accounts expenditure identity. There is no estimation involved in this equation, so instead of including the equation via a link to an external object, we merely add the equation as inline text.

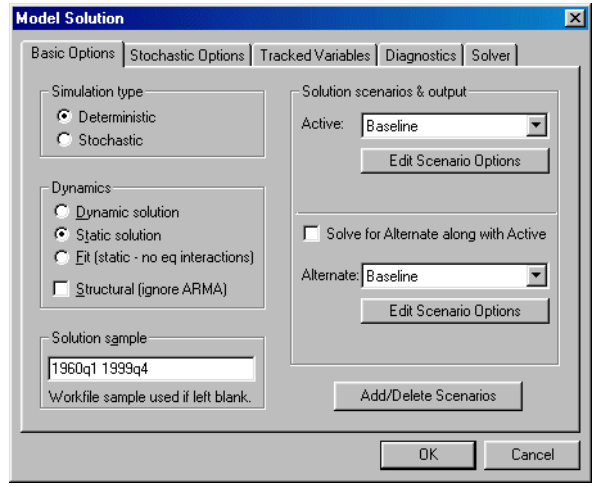
To add the identity, we click with the right mouse button anywhere in the equation window, and select **Insert…**. A dialog box will appear titled **Model Source Edit** which contains a text box with the heading **Enter one or more lines**. Simply type the identity, "Y = CN + I + G", into the text box, then click on **OK** to add it to the model.

The equation should now appear in the model window. The appearance differs slightly from the other equations, which is an indicator that the new equation is an inline text equation rather than a link.


Model: MODEL1 Workfile: MACROMOD\Macromod
View Proc Object Print Name Freeze Solve Equations Variables Text
Equations: 4
Scenario 1
eqi
eqcn
eqr
"y = cn + i + g"
Eq1: i = F(r, y)
Eq2: cn = F(cn, y)
Eq3: r = F(m, r, y)
Eq4: y = F(cn, g, i)

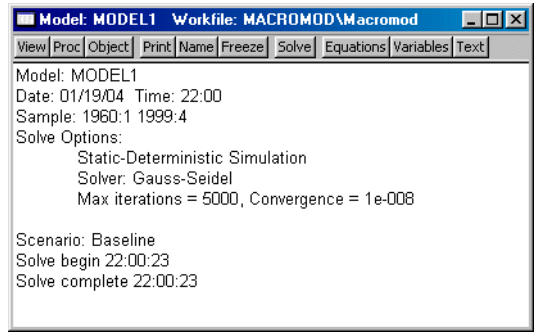
Our model specification is now complete. At this point, we can proceed straight to solving the model. To solve the model, simply click on the **Solve** button in the model window button bar.

There are many options available from the dialog, but for the moment we will consider only the basic settings. As our first exercise in assessing our model, we would like to examine the ability of our model to provide one-period ahead forecasts of our endogenous variables. To do this, we can look at the predictions of our model against our historical data, using actual values for both the exogenous and the lagged endogenous variables of the model. In EViews, we refer to this as a *static* simulation. We may easily perform this type of simulation by choosing **Static solution** in the **Dynamics** box of the dialog.



We must also adjust the sample over which to solve the model, so as to avoid initializing our solution with missing values from our data. Most of our series are defined over the range of 1947Q1 to 1999Q4, but our money supply series is available only from 1959Q1. Because of this, we set the sample to 1960Q1 to 1999Q4, allowing a few extra periods prior to the sample for any lagged variables.

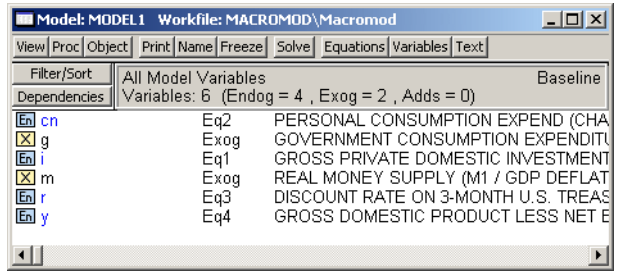
We are now ready to solve the model. Simply click on **OK** to start the calculations. The model window will switch to the **Solution Messages** view.



The output should be fairly self-explanatory. In this case, the solution took less than a second and there were no errors while performing the calculations.

Now that we have solved the model, we would like to look at the results. When we solved the model, the results for the endogenous variables were placed into series in the workfile with names determined by the name aliasing rules of the model. Since these series are ordinary EViews objects, we could use the workfile window to open the series and examine them directly. However, the model object provides a much more convenient way to work with the series through a view called the Variable View.

The easiest way to switch to the variable view is to click on the button labeled **Variables** on the model window button bar.

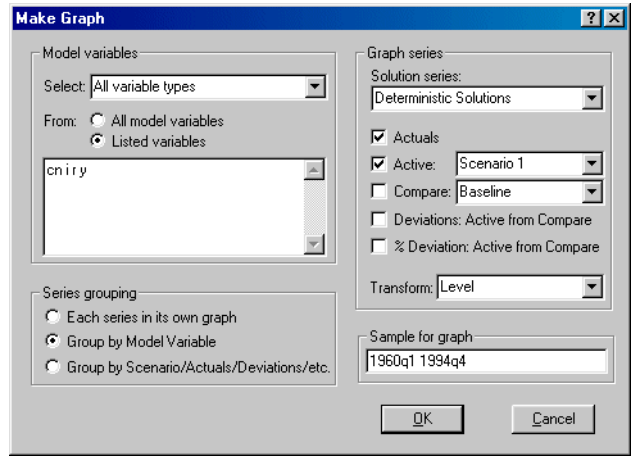

Model: MODEL1 Workfile: MACROMOD\Macromod
View Proc Object Print Name Freeze Solve Equations Variables Text
Filter/Sort
Dependencies
All Model Variables
Variables: 6 (Endog = 4 , Exog = 2 , Adds = 0)
Baseline
cn Eq2 PERSONAL CONSUMPTION EXPEND (CHA
g Exog GOVERNMENT CONSUMPTION EXPENDITU
i Eq1 GROSS PRIVATE DOMESTIC INVESTMENT
m Exog REAL MONEY SUPPLY (M1 / GDP DEFLAT
r Eq3 DISCOUNT RATE ON 3-MONTH U.S. TREAS
y Eq4 GROSS DOMESTIC PRODUCT LESS NET E

In the variable view, each line in the window is used to represent a variable. The line contains an icon indicating the variable type (endogenous, exogenous or add factor), the name of the variable, the equation with which the variable is associated (if any), and the description field from the label of the underlying series (if available). The name of the variable may be colored according to its status, indicating whether it is being traced (blue) or whether it has been overridden (red). In our model, we can see from the variable view that CN, I, R and Y are endogenous variables in the model, while G and M are exogenous.

Much of the convenience of the variable view comes from the fact that it allows you to work directly with the names of the variables in the model, rather than the names of series in the workfile. This is useful because when working with a model, there will often be many different series associated with each variable. For endogenous variables, there will be the actual historical values and one or more series of solution values. For exogenous variables, there may be several alternative scenarios for the variable. The variable view

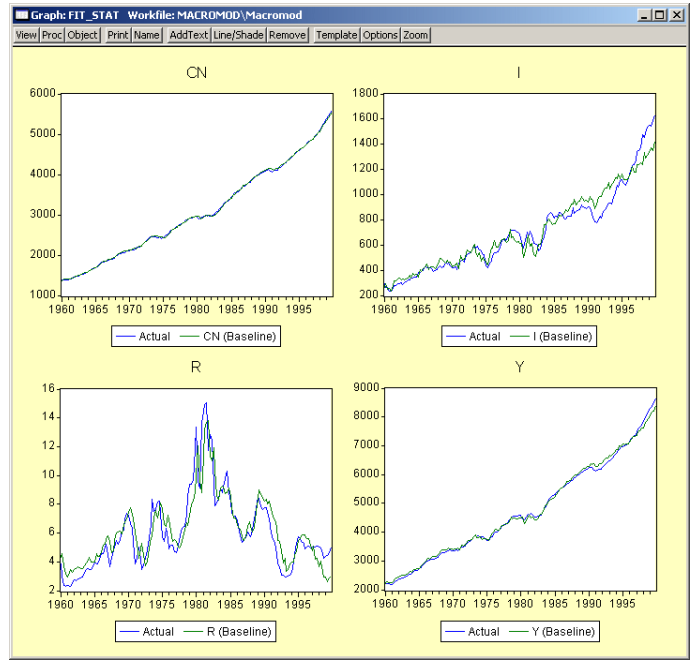
and its associated procedures help you move between these different sets of series without having to worry about the many different names involved.

For example, to look at graphs containing the actual and fitted values for the endogenous variables in our model, we simply select the four variables (by holding down the control key and clicking on the variable names), then use **Proc/ Make Graph...** to enter the dialog.



Again, the dialog has many options, but for our current purposes, we can leave most settings at their default values. Simply make sure that the **Actuals** and **Active** checkboxes are checked, set the sample for the graph to 1960Q1 to 1999Q4, then click on **OK**.

The graphs show that as a one-step ahead predictor, the model performs quite well, although the ability of the model to predict investment deteriorates during the second half of the sample.

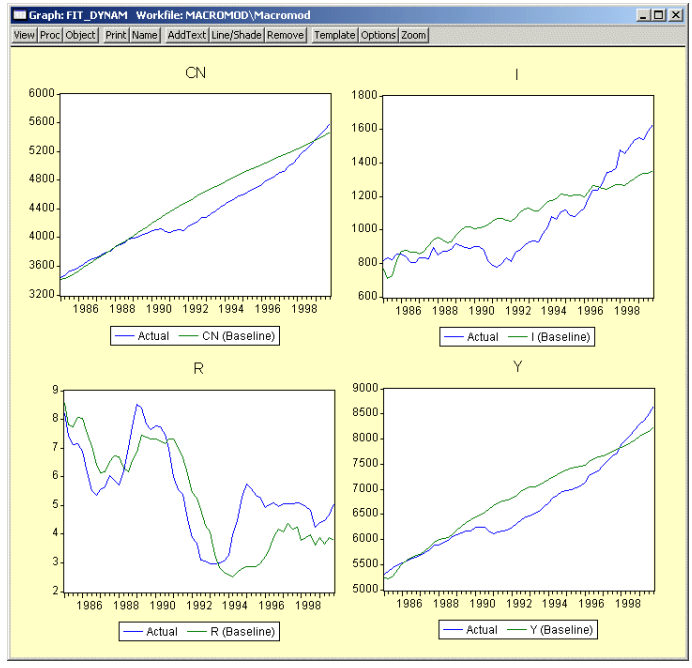


An alternative way of evaluating the model is to examine how the model performs when used to forecast many periods into the future. To do this, we must use our forecasts from previous periods, not actual historical data, when assigning values

to the lagged endogenous terms in our model. In EViews, we refer to such a forecast as a *dynamic* forecast.

To perform a dynamic forecast, we resolve the model with a slightly different set of options. Return to the model window and again click on the **Solve** button. In the model solution dialog, choose **Dynamic solution** in the **Dynamics** section of the dialog, and set the solution sample to 1985Q1 to 1999Q4.

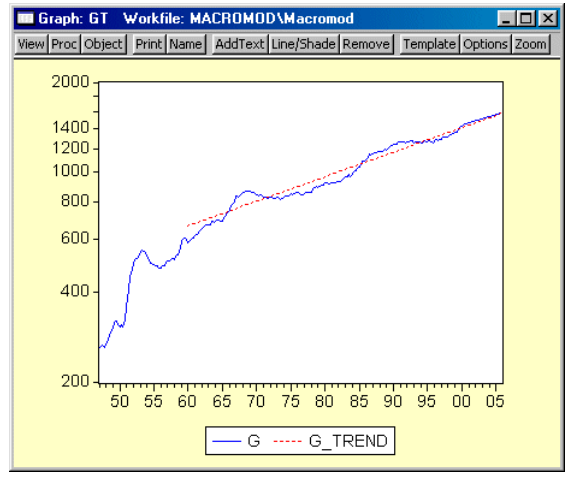
Click on **OK** to solve the model. To examine the results, return to the variable view, select the endogenous series again, and use **Proc/Make Graph…** exactly as above. Make sure the sample is set to 1985Q1 to 1999Q4 then click on **OK**. The results illustrate how our model would have performed if we had used it back in 1985 to make a forecast for the economy over the next fifteen years, assuming that we had used the correct paths for the exogenous variables (in reality, we would not have known these values at the time the forecasts were generated). Not surprisingly, the results show substantial deviations from the actual outcomes, although they do seem to follow the general trends in the data.



Once we are satisfied with the performance of our model against historical data, we can use the model to forecast future values of our endogenous variables. The first step in producing such a forecast is to decide on values for our exogenous variables during the forecast period. These may be based on our best guess as to what will actually happen, or they may be simply one particular possibility that we are interested in considering. Often we will be interested in constructing several different paths and then comparing the results.

In our model, we must provide future values for our two exogenous variables: government expenditure (G), and the real money supply (M). For our example, we will try to construct a set of paths that broadly follow the trends of the historical data.

A quick look at our historical series for G suggests that the growth rate of G has been fairly constant since 1960, so that the log of G roughly follows a linear trend. Where G deviates from the trend, the deviations seem to follow a cyclical pattern.

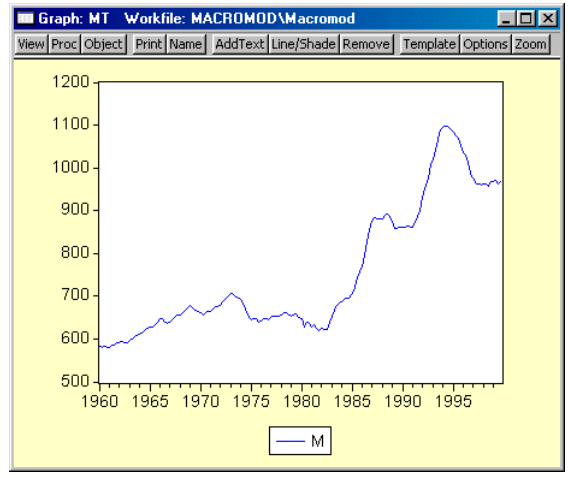


As a simple model of this behavior, we can regress the log of G against a constant and a time trend, using an AR(4) error structure to model the cyclical deviations. This gives the following equation, which we save in the workfile as EQG:

```
log(g) = 6.252335363 + 0.004716422189*@trend +
   [ar(1)=1.169491542,ar(2)=0.1986105964,ar(3)=0.239913126,ar(
   4)=-0.2453607091]
```

To produce a set of future values for G, we can use this equation to perform a dynamic forecast from 2000Q1 to 2005Q4, saving the results back into G itself (see page 797 for details).

The historical path of the real M1 money supply, M, is quite different from G, showing spurts of growth followed by periods of stability. For now, we will assume that the real money supply simply remains at its last observed historical value over the entire forecast period.

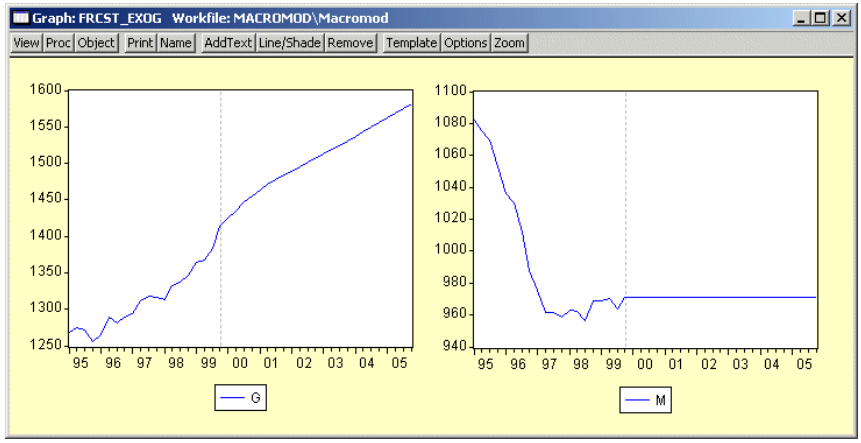


We can use an EViews series statement to fill in this path. The following lines will fill the series M from 2000Q1 to the last observation in the sample with the last observed historical value for M:

```
smpl 2000q1 @last
series m = m(-1)
```

```
smpl @all
```

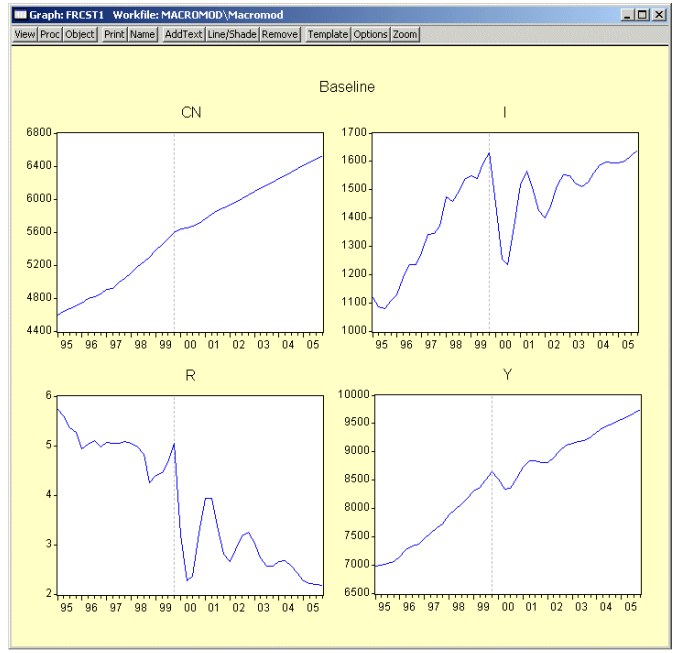
We now have a set of possible values for our exogenous variables over the forecast period. After plotting, we have:



To produce forecasts for our *endogenous* variables, we return to the model window, click on **Solve**, choose **Dynamic Solution**, set the forecast sample for 2000Q1 to 2005Q4, and then click on **OK**. The Solution Messages screen should appear, indicating that the model was successfully solved.

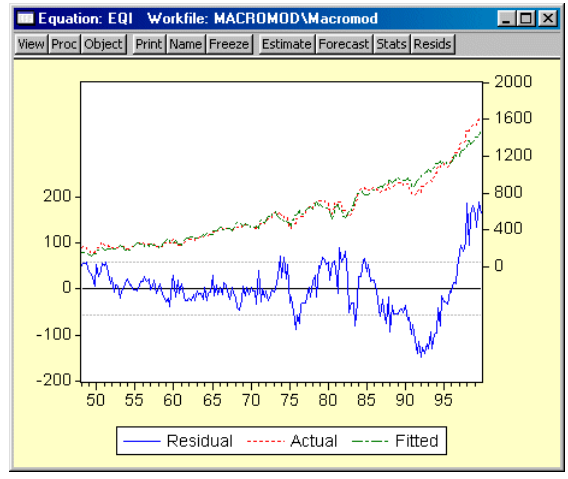
To examine the results in a graph, we again use **Proc/Make Graph…** from the variables view, set the sample to 1995Q1 to 2005Q4 (so that we include five years of historical data), then click on **OK**. After adding a line in 1999Q4 to separate historical and actual results, we get a graph showing the results:

We observe strange behavior in the results. At the beginning of the forecast period, we see a heavy dip in investment, GDP, and interest rates. This is followed by a series of oscillations in these series with a period of about a year, which die out slowly during the forecast period. This is not a particularly convincing forecast.



There is little in the paths of our exogenous variables or the history of our endogenous variables that would lead to this sharp dip, suggesting that the problem may lie with the residuals of our equations. Our investment equation is the most likely candidate, as it has a large, persistent positive residual near the end of the historical data (see figure below). This residual will be set to zero over the forecast period when solving the model, which might be the cause of the sudden drop in investment at the beginning of the forecast.

One way of dealing with this problem would be to change the specification of the investment equation. The simplest modification would be to add an autoregressive component to the equation, which would help reduce the persistence of the error. A better alternative would be to try to modify the variables in the equation so that the equation can provide some explanation for the sharp rise in investment during the 1990s.



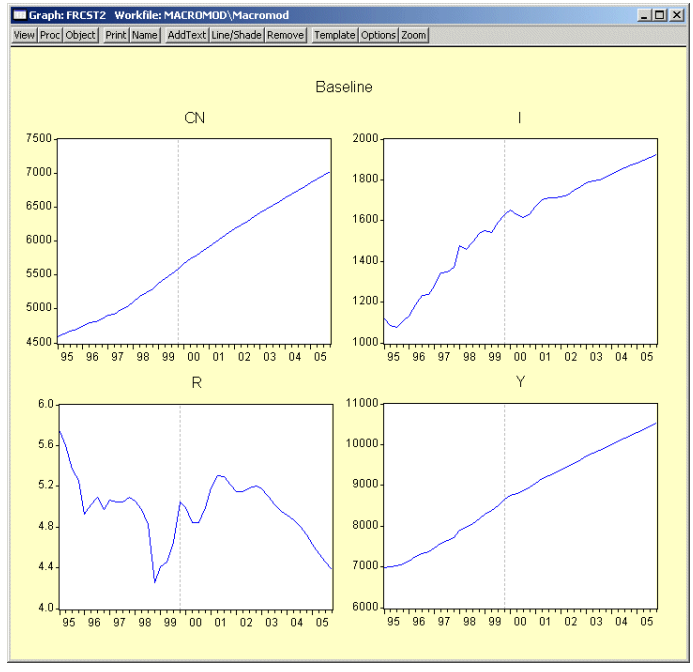
An alternative approach to the problem is to leave the equation as it is, but to include an add factor in the equation so that we can model the path of the residual by hand. To include the add factor, we switch to the equation view of the model, double click on the investment equation, EQI, select the **Add factor** tab. Under **Factor type**, choose **Equation intercept (residual shift)**. A prompt will appear asking if we would like to create the add factor series. Click on **Yes** to create the series. When you return to the variable view, you should see that a new variable, I_A, has been added to the list of variables in the model.

Using the add factor, we can specify any path we choose for the residual of the investment equation during the forecast period. By examining the **Actual/Fitted/Residual Graph** view from the equation object, we see that near the end of the historical data, the residual appears to be hovering around a value of about 160. We will assume that this value holds throughout the forecast period. We can set the add factor using a few simple EViews commands:

```
smpl 2000q1 @last
i_a = 160
smpl @all
```

With the add factor in place, we can follow exactly the same procedure that we followed above to produce a new set of solutions for the model and a new graph for the results.

Including the add factor in the model has made the results far more appealing. The sudden dip in the first period of the forecast that we saw above has been removed. The oscillations are still apparent, but are much less pronounced.



So far, we have been working under the assumption that our stochastic equations hold exactly over the forecast period. In reality, we would expect to see the same

sort of errors occurring in the future as we have seen in history. We have also been ignoring the fact that some of the coefficients in our equations are estimated, rather than fixed at known values. We may like to reflect this uncertainty about our coefficients in some way in the results from our model.

We can incorporate these features into our EViews model using stochastic simulation.

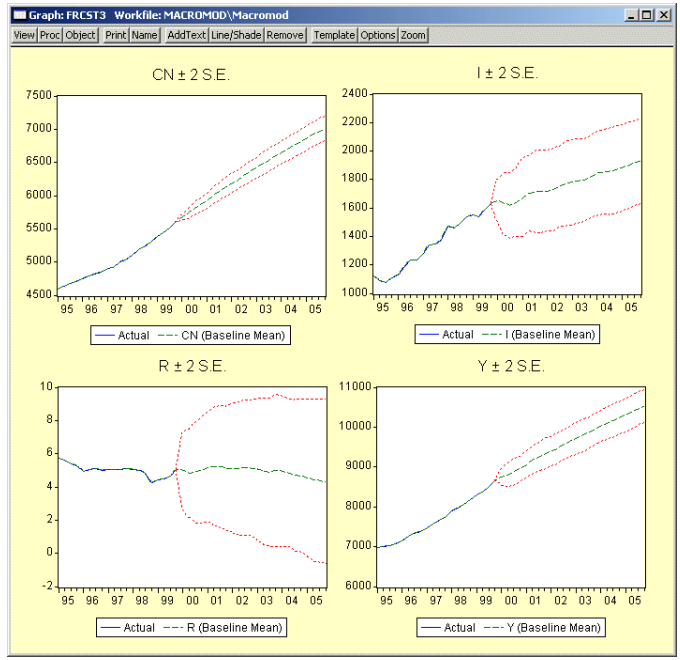
Up until now, we have thought of our model as forecasting a single point for each of our endogenous variables at each observation. As soon as we add uncertainty to the model, we should think instead of our model as predicting a whole distribution of outcomes for each variable at each observation. Our goal is to summarize these distributions using appropriate statistics.

If the model is linear (as in our example) and the errors are normal, then the endogenous variables will follow a normal distribution, and the mean and standard deviation of each distribution should be sufficient to describe the distribution completely. In this case, the mean will actually be equal to the deterministic solution to the model. If the model is not linear, then the distributions of the endogenous variables need not be normal. In this case, the quantiles of the distribution may be more informative than the first two moments, since the distributions may have tails which are very different from the normal case. In a non-linear model, the mean of the distribution need not match up to the deterministic solution of the model.

EViews makes it easy to calculate statistics to describe the distributions of your endogenous variables in an uncertain environment. To simulate the distributions, the model object uses a Monte Carlo approach, where the model is solved many times with pseudo-random numbers substituted for the unknown errors at each repetition. This method provides only approximate results. However, as the number of repetitions is increased, we would expect the results to approach their true values.

To return to our simple macroeconomic model, we can use a stochastic simulation to provide some measure of the uncertainty in our results by adding error bounds to our predictions. From the model window, click on the **Solve** button. When the model solution dialog appears, choose **Stochastic** for the simulation type. In the **Solution scenarios & output** box, make sure that the **Std. Dev.** checkbox in the **Active** section is checked. Click on **OK** to begin the simulation.

The simulation should take about half a minute. Status messages will appear to indicate progress through the repetitions. When the simulation is complete, you may return to the variable view, use the mouse to select the variables as discussed above, and then select **Proc/Make Graph...**. When the **Make Graph** dialog appears, select the option **Mean +- 2 standard deviations** in the **Solution Series** list box in the **Graph Series** area on the right of the dialog. Set the sample to 1995Q1 to 2005Q4 and click on **OK**.

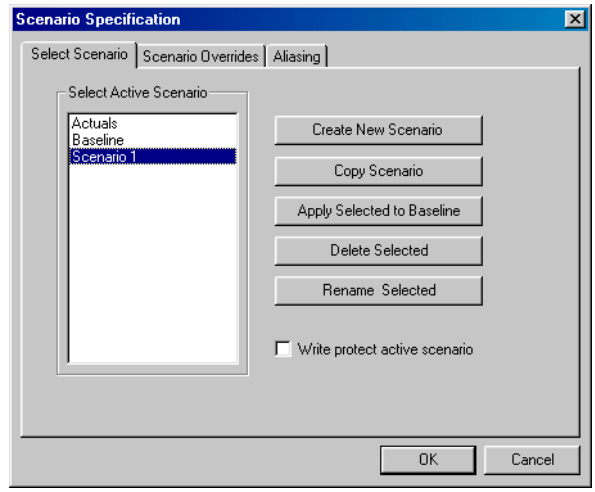

Graph: FRCST3 Workfile: MACROMOD\Macromod
View Proc Object Print Name AddText Line/Shade Remove Template Options Zoom
CN ± 2 S.E.
I ± 2 S.E.
R ± 2 S.E.
Y ± 2 S.E.
Actual
CN (Baseline Mean)
I (Baseline Mean)
R (Baseline Mean)
Y (Baseline Mean)

The error bounds in the resulting output graph show that we should be reluctant to place too much weight on the point forecasts of our model, since the bounds are quite wide on several of the variables. Much of the uncertainty is probably due to the large residual in the investment equation, which is creating a lot of variation in investment and interest rates in the stochastic simulation.

Another exercise we might like to consider when working with our model is to examine how the model behaves under alternative assumptions with respect to the exogenous variables. One approach to this would be to directly edit the exogenous series so that they contain the new values, and then resolve the model, overwriting any existing results. The problem with this approach is that it makes it awkward to manage the data and to compare the different sets of outcomes.

EViews provides a better way of carrying out exercises such as this through the use of model scenarios. Using a model scenario, you can override a subset of the exogenous variables in a model to give them new values, while using the values stored in the actual series for the remainder of the variables. When you solve for a scenario, the values of the endogenous variables are assigned into workfile series with an extension specific to that scenario, making it easy to keep multiple solutions for the model within a single workfile.

To create a scenario, we begin by selecting **View/Scenarios...** from the model object menus. The scenario specification dialog will appear with a list of the scenarios currently included in the model. There are two special scenarios that are always present in the model: **Actuals** and **Baseline**. These two scenarios are special in that they cannot contain any overridden variables.



They differ in that the actuals scenario writes its solution values directly into the workfile series with the same names as the endogenous variables, while the baseline scenario writes its solution values back into workfile series with the extension "_0".

To add a new scenario to the model, simply click on the button labeled **Create new scenario**. A new scenario will be created immediately. You can use this dialog to select which scenario is currently active, or to rename and delete scenarios.

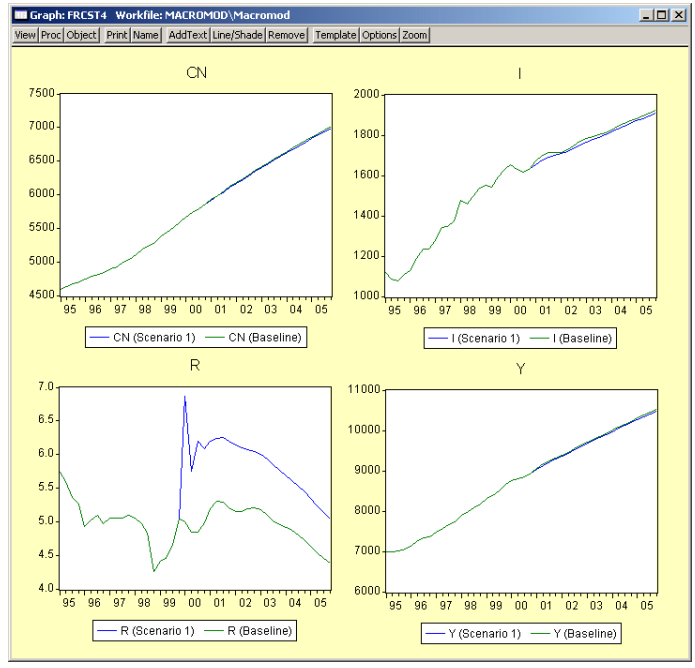
Once we have created the scenario, we can modify the scenario from the baseline case by overriding one of our exogenous variables. To do this, return to the variable window of the model, click on the variable M, use the right mouse button to call up the **Properties** dialog for the variable, and then in the **Scenario** box, click on the checkbox for **Use override series in scenario**. A message will appear asking if you would like to create the new series. Click on **Yes** to create the series, then **OK** to return to the variable window.

In the variable window, the variable name "M" should now appear in red, indicating that it has been overridden in the active scenario. This means that the variable M will now be bound to the series M_1 instead of the series M when solving the model. In our previous forecast for M, we assumed that the real money supply would be kept at a constant level during the forecast period. For our alternative scenario, we are going to assume that the real money supply is contracted sharply at the beginning of the forecast period, and held at this lower value throughout the forecast. We can set the new values using a few simple commands:

```
smpl 2000q1 2005q4
series m_1 = 900
smpl @all
```

As before, we can solve the model by clicking on the **Solve** button. Restore the **Simulation type** to deterministic, make sure that Scenario 1 is the active scenario, then click on **OK**. Once the solution is complete, we can use **Proc/Make Graph...** to display the results following the same procedure as above. Restore the **Solution series** list box to the setting **Deterministic solutions**, then check both the **Active** and **Compare** solution checkboxes

below, making sure that the active scenario is set to Scenario 1, and the comparison scenario is set to Baseline. Again set the sample to 1995Q1 to 2005Q4. The following graph should be displayed:



The simulation results suggest that the cut in the money supply causes a substantial increase in interest rates, which creates a small reduction in investment and a relatively minor drop in income and consumption. Overall, the predicted effects of changes in the money supply on the real economy are relatively minor in this model.

This concludes the discussion of our example model. The remainder of this chapter provides detailed information about working with particular features of the EViews model object.

Building a Model

Creating a Model

The first step in working with a model is to create the model object itself. There are several different ways of creating a model:

- You can create an empty model by using **Object/New Object...** and then choosing **Model**, or by performing the same operation using the right mouse button menu from inside the workfile window.
- You can select a list of estimation objects in the workfile window (equations, VARs, systems), and then select **Open as Model** from the right mouse button menu. This item will create a model which contains the equations from the selected objects as links.

- You can use the **Make model** procedure from an estimation object to create a model containing the equation or equations in that object.

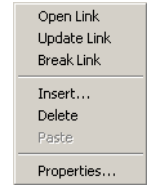
Adding Equations to the Model

The equations in a model can be classified into two types: linked equations and inline equations. Linked equations are equations that import their specification from other objects in the workfile. Inline equations are contained inside the model as text.

There are a number of ways to add equations to your model:

- To add a linked equation: from the workfile window, select the object which contains the equation or equations you would like to add to the model, then copy-and-paste the object into the model equation view window.

- To add an equation using text: select **Insert…** from the right mouse button menu. In the text box titled: **Enter one or more lines…,** type in one or more equations in standard EViews format. You can also add linked equations from this dialog by typing a colon followed by the name of the object you would like to link to, for example ":EQ1", because this is the text form of a linked object.



In an EViews model, the first variable that appears in an equation will be considered the endogenous variable for that equation. Since each endogenous variable can be associated with only one equation, you may need to rewrite your equations to ensure that each equation begins with a different variable. For example, say we have an equation in the model:

```
x / y = z
```

EViews will associate the equation with the variable X. If we would like the equation to be associated with the variable Y, we would have to rewrite the equation:

```
1 / y * x = z
```

Note that EViews has the ability to handle simple expressions involving the endogenous variable. You may use functions like LOG, D, and DLOG on the left-hand side of your equation. EViews will normalize the equation into explicit form if the Gauss-Seidel method is selected for solving the model.

Removing equations from the model

To remove equations from the model, simply select the equations using the mouse in Equation view, then use **Delete** from the right mouse button menu to remove the equations.

Both adding and removing equations from the model will change which variables are considered endogenous to the model.

Updating Links in the Model

If a model contains linked equations, changes to the specification of the equations made outside the model can cause the equations contained in the model to become out of date. You can incorporate these changes in the model by using **Proc/Link/Update All Links**. Alternatively, you can update just a single equation using the **Proc/Link/Update Link** item from the right mouse button menu. Links are also updated when a workfile is reloaded from disk.

Sometimes, you may want to sever equations in the model from their linked objects. For example, you may wish to see the entire model in text form, with all equations written in place. To do this, you can use the **Proc/Link/Break All Links** procedure to convert all linked equations in the model into inline text. You can convert just a single equation by selecting the equation, then using **Break Link** from the right mouse button menu.

When a link is broken, the equation is written in text form with the unknown coefficients replaced by their point estimates. Any information relating to uncertainty of the coefficients will be lost. This will have no effect on deterministic solutions to the model, but may alter the results of stochastic simulations if the **Include coefficient uncertainty** option has been selected.

Working with the Model Structure

As with other objects in EViews, we can look at the information contained in the model object in several ways. Since a model is a set of equations that describe the relationship between a set of variables, the two primary views of a model are the equation view and the variable view. EViews also provides two additional views of the structure of the model: the block view and the text view.

Equation View

The equation view is used for displaying, selecting, and modifying the equations contained in the model. An example of the equation view can be seen on page 783.

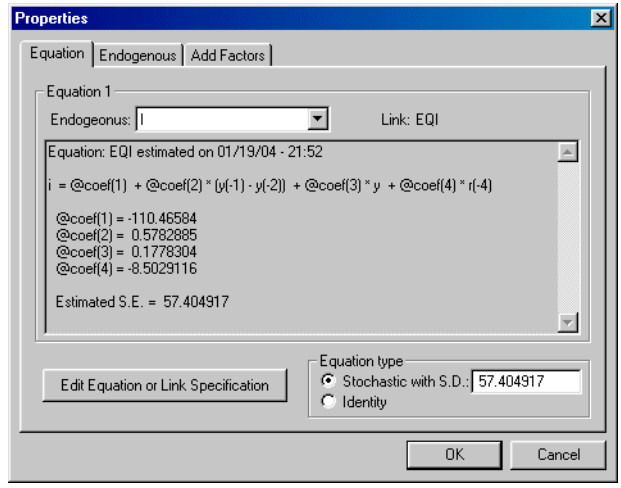
Each line of the window is used to represent either a linked object or an inline text equation. Linked objects will appear similarly to how they do in the workfile, with an icon representing their type, followed by the name of the object. Even if the linked object contains many equations, it will use only one line in the view. Inline equations will appear with a "TXT" icon, followed by the beginning of the equation text in quotation marks.

The remainder of the line contains the equation number, followed by a symbolic representation of the equation, indicating which variables appear in the equation.

Any errors in the model will appear as red lines containing an error message describing the cause of the problem.

You can open any linked objects directly from the equation view. Simply select the line representing the object using the mouse, then choose **Open Link** from the right mouse button menu.

The contents of a line can be examined in more detail using the equation properties dialog. Simply select the line with the mouse, then choose **Properties...** from the right mouse button menu. Alternatively, simply double click on the object to call up the dialog.



For a link to a single equation, the dialog shows the functional form of the equation, the values of any estimated coefficients, and the standard error of the equation residual from the estimation. If the link is to an object containing many equations, you can move between the different equations imported from the object using the **Endogenous** list box at the top of the dialog. For an inline equation, the dialog simply shows the text of the equation.

The **Edit Equation or Link Specification** button allows you to edit the text of an inline equation or to modify a link to point to an object with a different name. A link is represented in text form as a colon followed by the name of the object. Note that you cannot modify the specification of a linked object from within the model object, you must work directly with the linked object itself.

In the bottom right of the dialog, there are a set of fields that allow you to set the stochastic properties of the residual of the equation. If you are only performing deterministic simulations, then these settings will not affect your results in any way. If you are performing stochastic simulations, then these settings are used in conjunction with the solution options to determine the size of the random innovations applied to this equation.

The **Stochastic with S.D.** option for **Equation type** lets you set a standard deviation for any random innovations applied to the equation. If the standard deviation field is blank or is set to "NA", then the standard deviation will be estimated from the historical data. The **Identity** option specifies that the selected equation is an identity, and should hold without error even in a stochastic simulation. See "Stochastic Options" on page 810 below for further details.

The equation properties dialog also gives you access to the property dialogs for the endogenous variable and add factor associated with the equation. Simply click on the appropriate tab. These will be discussed in greater detail below.

Variable View

The variable view is used for adjusting options related to variables and for displaying and editing the series associated with the model (see the discussion in "An Example Model" (p. 784)). The variable view lists all the variables contained in the model, with each line representing one variable. Each line begins with an icon classifying the variable as endogenous, exogenous or an add factor. This is followed by the name of the variable, the equation number associated with the variable, and the description of the variable. The description is read from the associated series in the workfile.

Note that the names and types of the variables in the model are determined fully by the equations of the model. The only way to add a variable or to change the type of a variable in the model is to modify the model equations.

You can adjust what is displayed in the variable view in a number of ways. By clicking on the **Filter/Sort** button just above the variable list, you can choose to display only variables that match a certain name pattern, or to display the variables in a particular order. For example, sorting by type of variable makes the division into endogenous and exogenous variables clearer, while sorting by override highlights which variables have been overridden in the currently active scenario.

The variable view also allows you to browse through the dependencies between variables in the model by clicking on the **Dependencies** button. Each equation in the model can be thought of as a set of links that connect other variables in the model to the endogenous variable of the equation. Starting from any variable, we can travel up the links, showing all the endogenous variables that this variable directly feeds into, or we can travel down the links, showing all the variables upon which this variable directly depends. This may sometimes be useful when trying to find the cause of unexpected behavior. Note, however, that in a simultaneous model, every endogenous variable is indirectly connected to every other variable in the same block, so that it may be hard to understand the model as a whole by looking at any particular part.

You can quickly view or edit one or more of the series associated with a variable by double clicking on the variable. For several variables, simply select each of them with the mouse then double click inside the selected area.

Block Structure View

The block structure view of the model analyzes and displays any block structure in the dependencies of the model.

Block structure refers to whether the model can be split into a number of smaller parts, each of which can be solved for in sequence. For example, consider the system:

block 1	x = y + 4
	y = 2*x – 3
block 2	z = x + y

Because the variable Z does not appear in either of the first two equations, we can split this equation system into two blocks: a block containing the first two equations, and a block containing the third equation. We can use the first block to solve for the variables X and Y, then use the second block to solve for the variable Z. By using the block structure of the system, we can reduce the number of variables we must solve for at any one time. This typically improves performance when calculating solutions.

Blocks can be classified further into *recursive* and *simultaneous* blocks. A recursive block is one which can be written so that each equation contains only variables whose values have already been determined. A recursive block can be solved by a single evaluation of all the equations in the block. A simultaneous block cannot be written in a way that removes feedback between the variables, so it must be solved as a simultaneous system. In our example above, the first block is simultaneous, since X and Y must be solved for jointly, while the second block is recursive, since Z depends only on X and Y, which have already been determined in solving the first block.

The block structure view displays the structure of the model, labeling each of the blocks as recursive or simultaneous. EViews uses this block structure whenever the model is solved. The block structure of a model may also be interesting in its own right, since reducing the system to a set of smaller blocks can make the dependencies in the system easier to understand.

Text View

The text view of a model allows you to see the entire structure of the model in a single screen of text. This provides a quick way to input small models, or a way to edit larger models using copy-and-paste.

The text view consists of a series of lines. In a simple model, each line simply contains the text of one of the inline equations of the model. More complicated models may contain one of more of the following:

- A line beginning with a colon ":" represents a link to an external object. The colon must be followed by the name of an object in the workfile. Equations contained in the external object will be imported into the model whenever the model is opened, or when links are updated.

- A line beginning with "@ADD" specifies an add factor. The add factor command has the form:

  ```
  @add(v) endogenous_name add_name
  ```

  where endogenous_name is the name of the endogenous variable of the equation to which the add factor will be applied, and add_name is the name of the series. The option (v) is used to specify that the add factor should be applied to the endogenous variable. The default is to apply the add factor to the residual of the equation. See "Using Add Factors" on page 802 for details.

- A line beginning with "@INNOV"' specifies an innovation variance. The innovation variance has two forms. When applied to an endogenous variable it has the form:

  ```
  @innov endogenous_name number
  ```

  where endogenous name is the name of the endogenous variable and number is the standard deviation of the innovation to be applied during stochastic simulation. When applied to an exogenous variable, it has the form:

  ```
  @innov exogenous_name number_or_series
  ```

  where exogenous name is the name of the exogenous variable and number_or_series is either a number or the name of the series that contains the standard deviation to be applied to the variable during stochastic simulation. Note that when an equation in a model is linked to an external estimation object, the variance from the estimated equation will be brought into the model automatically and does not require an `@innov` specification unless you would like to modify its value.

- The keyword "@TRACE", followed by the names of the endogenous variables that you wish to trace, may be used to request model solution diagnostics. See "Diagnostics" on page 813.

Users of earlier versions of EViews should note that two commands that were previously available, `@assign` and `@exclude`, are no longer part of the text form of the model. These commands have been removed because they now address options that apply only to specific model scenarios rather than to the model as a whole. When loading in models created by earlier versions of EViews, these commands will be converted automatically into scenario options in the new model object.

Specifying Scenarios

When working with a model, you will often want to compare model predictions under a variety of different assumptions regarding the paths of your exogenous variables, or with one or more of your equations excluded from the model. Model scenarios allow you to do this without overwriting previous data or changing the structure of your model.

The most important function of a scenario is to specify which series will be used to hold the data associated with a particular solution of the model. To distinguish the data associated with different scenarios, each scenario modifies the names of the model variables according to an aliasing rule. Typically, aliasing will involve adding an underline followed by a number, such as "_0" or "_1" to the variable names of the model. The data for each scenario will be contained in series in the workfile with the aliased names.

Model scenarios support the analysis of different assumptions for exogenous variables by allowing you to override a set of variables you would like to alter. Exogenous variables which are overridden will draw their values from series with names aliased for that scenario, while exogenous variables which are not overridden will draw their values from series with the same name as the variable.

Scenarios also allow you to exclude one or more endogenous variables from the model. When an endogenous variable is excluded, the equation associated with that variable is dropped from the model and the value of the variable is taken directly from the workfile series with the same name. Excluding an endogenous variable effectively treats the variable as an exogenous variable for the purposes of solving the model.

When excluding an endogenous variable, you can specify a sample range over which the variable should be excluded. One use of this is to handle the case where more recent historical data is available for some of your endogenous variables than others. By excluding the variables for which you have data, your forecast can use actual data where possible, and results from the model where data are not yet available.

Each model can contain many scenarios. You can view the scenarios associated with the current model by choosing **View/Scenario Specification...**as shown above on page 793.

There are two special scenarios associated with every model: actuals and baseline. These two scenarios have in common the special property that they cannot contain any overrides or excludes. They differ in that the actuals scenario writes the values for endogenous variables back into the series with the same name as the variables in the model, while the baseline scenario modifies the names. When solving the model using actuals as your active scenario, you should be careful not to accidentally overwrite your historical data.

The baseline scenario gets its name from the fact that it provides the base case from which other scenarios are constructed. Scenarios differ from the baseline by having one or more variables overridden or excluded. By comparing the results from another scenario against those of the baseline case, we can separate out the movements in the endogenous variables that are due to the changes made in that particular scenario from movements which are present in the baseline itself.

The **Select Scenario** page of the dialog allows you to select, create, copy, delete and rename the scenarios associated with the model. You may also apply the selected scenario

to the baseline data, which involves copying the series associated with any overridden variables in the selected scenario on top of the baseline values. Applying a scenario to the baseline is a way of committing to the edited values of the selected scenario making them a permanent part of the baseline case.

The **Scenario overrides** page provides a summary of variables which have been overridden in the selected scenario and equations which have been excluded. This is a useful way of seeing a complete list of all the changes which have been made to the scenario from the baseline case.

The **Aliasing** page allows you to examine the name aliasing rules associated with any scenario. The page displays the complete set of aliases that will be applied to the different types of variables in the model.

Although the scenario dialog lets you see all the settings for a scenario in one place, you will probably alter most scenario settings directly from the variable view instead. For both exogenous variables and add factors, you can select the variable from the variable view window, then use the right mouse button menu to call up the properties page for the variable. The override status of the variable can be adjusted using the **Use override** checkbox. Once a variable has been overridden, it will appear in red in the variable view.

Using Add Factors

Normally, when a model is solved deterministically, the equations of the model are solved so that each of the equations of the model is exactly satisfied. When a model is solved stochastically, random errors are added to each equation, but the random errors are still chosen so that their average value is zero.

If we have no information as to the errors in our stochastic equations that are likely to occur during the forecast period, then this behavior is appropriate. If, however, we have additional information as to the sort of errors that are likely during our forecast period, then we may incorporate that information into the model using add factors.

The most common use for add factors is to provide a smoother transition from historical data into the forecast period. Typically, add factors will be used to compensate for a poor fit of one or more equations of the model near the end of the historical data, when we suspect this will persist into the forecast period. Add factors provide an ad hoc way of trying to adjust the results of the model without respecifying or reestimating the equations of the model.

In reality, an add factor is just an extra exogenous variable which is included in the selected equation in a particular way. EViews allows an add factor to take one of two forms. If our equation has the form:

$$f(y_i) = f_i(y, x) \tag{26.3}$$

then we can provide an add factor for the equation intercept or residual by simply including the add factor at the end of the equation:

$$f(y_i) = f_i(y, x) + a \tag{26.4}$$

Alternatively, we may provide an add factor for the endogenous variable of the model by using the add factor as an offset:

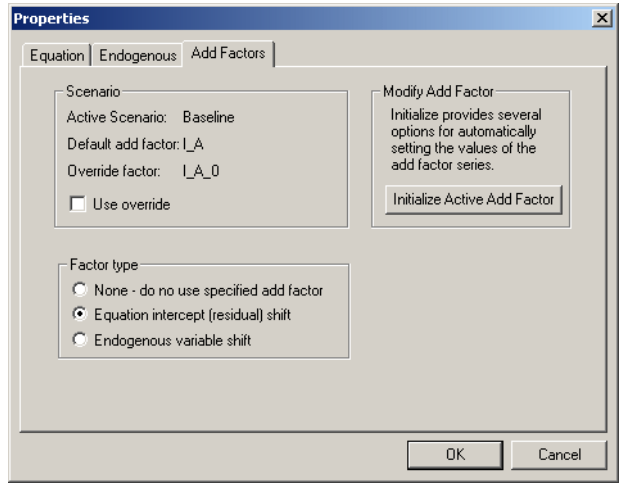
$$f(y_i - a) = f_i(y, x) \tag{26.5}$$

where the sign of the add factor is reversed so that it acts in the same direction as for the previous case.

If the endogenous variable appears by itself on the left hand side of the equal sign, then the two types of add factor are equivalent. If the endogenous variable is contained in an expression, for example, a log transformation, then this is no longer the case. Although the two add factors will have a similar effect, they will be expressed in different units with the former in the units of the residual of the equation, and the latter in the units of the endogenous variable of the equation.

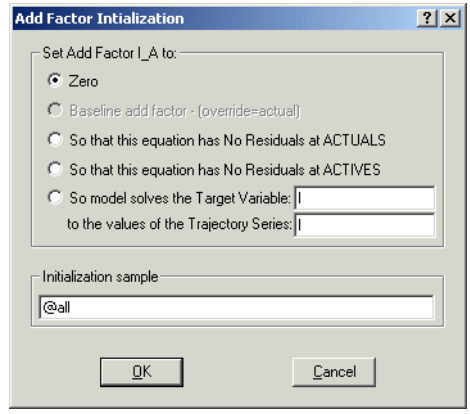
There are two ways to include add factors. The easiest way is to go to the equation view of the model, then double click on the equation in which you would like to include an add factor.

When the equation properties dialog appears, switch to the **Add Factors** tab. In the **Factor type** box, select whether you would like an intercept or an endogenous variable shift add factor. A message box will prompt for whether you would like to create a series in the workfile to hold the add factor values. Click on **Yes** to create the series.



The series will initially be filled with NAs. You can initialize the add factor using one of several methods by clicking on the **Initialize Add Factor** button.

A dialog box will come up offering the following options:


Add Factor Intialization
Set Add Factor I_A to:
Zero
Baseline add factor - (override=actual)
So that this equation has No Residuals at ACTUALS
So that this equation has No Residuals at ACTIVES
So model solves the Target Variable:
to the values of the Trajectory Series:
Initialization sample
@all
OK
Cancel

- **Zero**: set the add factor to zero for every period.
- **So that this equation has no residuals at actuals**: set the values of the add factor so that the equation is exactly satisfied without error when the variables of the model are set to the values contained in the actual series (typically the historical data).
- **So that this equation has no residuals at actives**: set the values of the add factor so that the equation is exactly satisfied without error when the variables of the model are set to the values contained in the endogenous and exogenous series associated with the active scenario.
- **So model solves the target variable to the values of the trajectory series**: set the values of the add factor so that an endogenous variable of the model follows a particular target path when the model is solved.

You can also change the sample over which you would like the add factor to be initialized by modifying the **Initialization sample** box. Click on OK to accept the settings.

Once an add factor has been added to an equation, it will appear in the variable view of the model as an additional variable. If an add factor is present in any scenario, then it must be present in every scenario, although the values of the add factor can be overridden for a particular scenario in the same way as for an exogenous variable.

The second way to handle add factors is to assign, initialize or override them for all the equations in the model at the same time using the **Proc/Add Factors** menu from the model window. For example, to create a complete set of add factors that make the model solve to actual values over history, we can use **Add Factors/Equation Assignment...** to create add factors for every equation, then use **Add Factors/Set Values...** to set the add factors so that all the equations have no residuals at the actual values.

When solving a model with an add factor, any missing values in the add factor will be treated as zeros.

Solving the Model

Once the model specification is complete, you can solve the model. EViews can perform both deterministic and stochastic simulations.

A deterministic simulation consists of the following steps:

- The block structure of the model is analyzed.
- The variables in the model are bound to series in the workfile, according to the override settings and name aliasing rules of the scenario that is being solved. If an endogenous variable is being tracked and a series does not already exist in the workfile, a new series will be created. If an endogenous variable is not being tracked, a temporary series will be created to hold the results.
- The equations of the model are solved for each observation in the solution sample, using an iterative algorithm to compute values for the endogenous variables.
- Any temporary series which were created are deleted.
- The results are rounded to their final values.

A stochastic simulation follows a similar sequence, with the following differences:

- When binding the variables, a temporary series is created for every endogenous variable in the model. Additional series in the workfile are used to hold the statistics for the tracked endogenous variables. If bounds are being calculated, extra memory is allocated as working space for intermediate results.
- The model is solved repeatedly for different draws of the stochastic components of the model. If coefficient uncertainty is included in the model, then a new set of coefficients is drawn before each repetition (note that coefficient uncertainty is ignored in nonlinear equations, or linear equations specified with PDL terms). During the repetition, errors are generated for each observation in accordance with the residual uncertainty and the exogenous variable uncertainty in the model. At the end of each repetition, the statistics for the tracked endogenous variables are updated to reflect the additional results.
- If a comparison is being performed with an alternate scenario, then the same set of random residuals and exogenous variable shocks are applied to both scenarios during each repetition. This is done so that the deviation between the two is based only on differences in the exogenous and excluded variables, not on differences in random errors.

Models Containing Future Values

So far, we have assumed that the structure of the model allows us to solve each period of the model in sequence. This will not be true in the case where the equations of the model contain future (as well as past) values of the endogenous variables.

Consider a model where the equations have the form:

$$F(y(-\text{maxlag}), \ldots, y(-1), y, y(1), \ldots, y(\text{maxlead}), x) = 0 \qquad (26.6)$$

where F is the complete set of equations of the model, y is a vector of all the endogenous variables, x is a vector of all the exogenous variables, and the parentheses follow the usual EViews syntax to indicate leads and lags.

Since solving the model for any particular period requires both past and future values of the endogenous variables, it is not possible to solve the model recursively in one pass. Instead, the equations from all the periods across which the model will be solved must be treated as a simultaneous system, and we will require terminal as well as initial conditions. For example, in the case with a single lead and a single lag and a sample that runs from s to t, we must effectively solve the entire stacked system:

$$\begin{aligned} F(y_{s-1}, y_s, y_{s+1}, x) &= 0 \\ F(y_s, y_{s+1}, y_{s+2}, x) &= 0 \\ F(y_{s+1}, y_{s+2}, y_{s+3}, x) &= 0 \\ &\dots \\ F(y_{t-2}, y_{t-1}, y_t, x) &= 0 \\ F(y_{t-1}, y_t, y_{t+1}, x) &= 0 \end{aligned} \tag{26.7}$$

where the unknowns are y_s, y_{s+1},... y_t the initial conditions are given by y_{s-1} and the terminal conditions are used to determine y_{t+1}. Note that if the leads or lags extend more than one period, we will require multiple periods of initial or terminal conditions.

To solve models such as these, EViews applies a Gauss-Seidel iterative scheme across all the observations of the sample. Roughly speaking, this involves looping repeatedly through every observation in the forecast sample, at each observation solving the model while treating the past and future values as fixed, where the loop is repeated until changes in the values of the endogenous variables between successive iterations become less than a specified tolerance.

This method is often referred to as the Fair-Taylor method, although the Fair-Taylor algorithm includes a particular handling of terminal conditions (the extended path method) that is slightly different from the options provided by EViews. When solving the model, EViews allows the user to specify fixed end conditions by providing values for the endogenous variables beyond the end of the forecast sample, or to determine the terminal conditions endogenously by adding extra equations for the terminal periods which impose either a constant level, a linear trend, or a constant growth rate on the endogenous variables for values beyond the end of the forecast period.

Although this method is not guaranteed to converge, failure to converge is often a sign of the instability which results when the influence of the past or the future on the present does not die out as the length of time considered is increased. Such instability is often undesirable for other reasons and may indicate a poorly specified model.

Model Consistent Expectations

One source of models in which future values of endogenous variables may appear in equations are models of economic behavior in which expectations of future periods influence the decisions made in the current period. For example, when negotiating long term wage contracts, employers and employees must consider expected changes in prices over the duration of the contract. Similarly, when choosing to hold a security denominated in foreign currency, an individual must consider how the exchange rate is expected to change over the time that they hold the security.

Although the way that individuals form expectations is obviously complex, if the model being considered accurately captures the structure of the problem, we might expect the expectations of individuals to be broadly consistent with the outcomes predicted by the model. In the absence of any other information, we may choose to make this relationship hold exactly. Expectations of this form are often referred to as *model consistent expectations*.

If we assume that there is no uncertainty in the model, imposing model consistent expectations simply involves replacing any expectations that appear in the model with the future values predicted by the model. In EViews, we can simply write out the expectation terms that appear in equations using the lead operator. A deterministic simulation of the model can then be run using EViews ability to solve models with equations which contain future values of the endogenous variables.

When we add uncertainty to the model, the situation becomes more complex. In this case, instead of the expectations of agents being set equal to the single deterministic outcome predicted by the model, the expectations of agents should be calculated based on the entire distribution of stochastic outcomes predicted by the model. To run a stochastic simulation of a model involving expectations would require a procedure like the following:

1. Take an initial guess as to a path for expectations over the forecast period (for example, by calculating a solution for the expectations in the deterministic case)

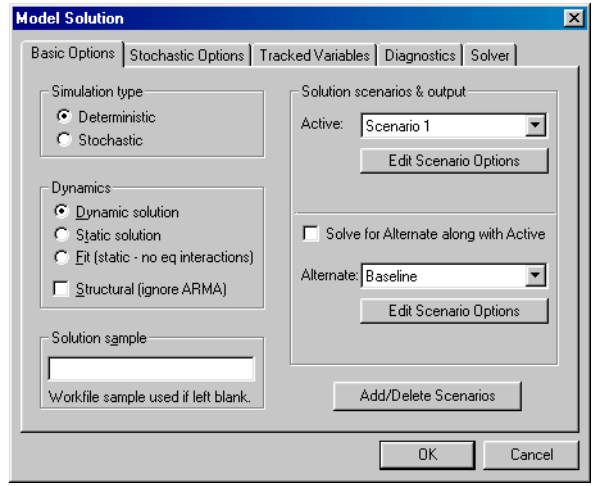
2. Run a large number of stochastic repetitions of the model holding these expectations constant, calculating the mean paths of the endogenous variables over the entire set of outcomes.

3. Test if the mean paths of the endogenous variables are equal to the current guess of expectations within some tolerance. If not, replace the current guess of expectations with the mean of the endogenous variables obtained in step 2, and return to step 2.

At present, EViews does not have built in functionality for automatically carrying out this procedure. Because of this, EViews will not perform stochastic simulations if your model contains equations involving future values of endogenous variables. We hope to add this functionality to future revisions of EViews.

Basic Options

To begin solving a model, you can use **Proc/Solve Model...** or you can simply click on the **Solve** button on the model toolbar. EViews will display a tabbed dialog containing the solution options.



The basic options page contains the most important options for the simulation. While the options on other pages can often be left at their default values, the options on this page will need to be set appropriately for the task you are trying to perform.

At the top left, the **Simulation type** box allows you to determine whether the model should be simulated deterministically or stochastically. In a deterministic simulation, all equations in the model are solved so that they hold without error during the simulation period, all coefficients are held fixed at their point estimates, and all exogenous variables are held constant. This results in a single path for the endogenous variables which can be evaluated by solving the model once.

In a stochastic simulation, the equations of the model are solved so that they have residuals which match to randomly drawn errors, and, optionally, the coefficients and exogenous variables of the model are also varied randomly (see "Stochastic Options" on page 810 for details). For stochastic simulation, the model solution generates a distribution of outcomes for the endogenous variables in every period. We approximate the distribution by solving the model many times using different draws for the random components in the model then calculating statistics over all the different outcomes.

Typically, you will first analyze a model using deterministic simulation, and then later proceed to stochastic simulation to get an idea of the sensitivity of the results to various sorts of error. You should generally make sure that the model can be solved deterministically and is behaving as expected before trying a stochastic simulation, since stochastic simulation can be very time consuming.

The next option is the **Dynamics** box. This option determines how EViews uses historical data for the endogenous variables when solving the model:

- When **Dynamic solution** is chosen, only values of the endogenous variables from before the solution sample are used when forming the forecast. Lagged endogenous variables and ARMA terms in the model are calculated using the solutions calculated in previous periods, not from actual historical values. A dynamic solution is typically the correct method to use when forecasting values several periods into the future (a multi-step forecast), or evaluating how a multi-step forecast would have performed historically.
- When **Static solution** is chosen, values of the endogenous variables up to the previous period are used each time the model is solved. Lagged endogenous variables and ARMA terms in the model are based on actual values of the endogenous variables. A static solution is typically used to produce a set of one-step ahead forecasts over the historical data so as to examine the historical fit of the model. A static solution cannot be used to predict more than one observation into the future.
- When the **Fit** option is selected, values of the endogenous variables for the current period are used when the model is solved. All endogenous variables except the one variable for the equation being evaluated are replaced by their actual values. The fit option can be used to examine the fit of each of the equations in the model when considered separately, ignoring their interdependence in the model. The fit option can only be used for periods when historical values are available for all the endogenous variables.

In addition to these options, the **Structural** checkbox gives you the option of ignoring any ARMA specifications that appear in the equations of the model.

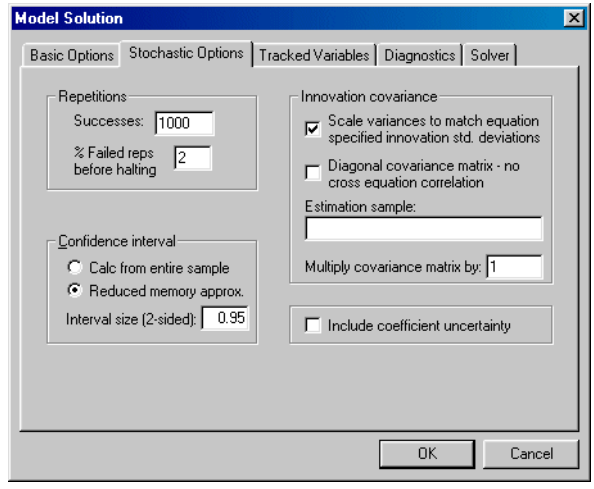
At the bottom left of the dialog is a box for the solution sample. The solution sample is the set of observations over which the model will be solved. Unlike in some other EViews procedures, the solution sample will not be contracted automatically to exclude missing data. For the solution to produce results, data must be available for all exogenous variables over the course of the solution sample. If you are carrying out a static solution or a fit, data must also be available for all endogenous variables during the solution sample. If you are performing a dynamic solution, only pre-sample values are needed to initialize any lagged endogenous or ARMA terms in the model.

On the right-hand side of the dialog are controls for selecting which scenarios we would like to solve. By clicking on one of the **Edit Scenario Options** buttons, you can quickly examine the settings of the selected scenario. The option **Solve for Alternate along with Active** should be used mainly in a stochastic setting, where the two scenarios must be solved together to ensure that the same set of random shocks is used in both cases. Whenever two models are solved together stochastically, a set of series will also be created containing the deviations between the scenarios (this is necessary because in a non-linear model, the difference of the means need not equal the mean of the differences).

When stochastic simulation has been selected, additional checkboxes are available for selecting which statistics you would like to calculate for your tracked endogenous variables. A series for the mean will always be calculated. You can also optionally collect series for the standard deviation or quantile bounds. Quantile bounds require considerable working memory, but are useful if you suspect that your endogenous variables may have skewed distributions or fat tails. If standard deviations or quantile bounds are chosen for either the active or alternate scenario, they will also be calculated for the deviations series.

Stochastic Options

The stochastic options page contains settings used during stochastic simulation. In many cases, you can leave these options at their default settings.



The **Repetitions** box, in the top left corner of the dialog, allows you to set the number of repetitions that will be performed during the stochastic simulation. A higher number of repetitions will reduce the sampling variation in the statistics being calculated, but will take more time. The default value of one thousand repetitions is generally adequate to get a good idea of the underlying values, although there may still be some random variation visible between adjacent observations.

Also in the repetitions box is a field labeled **% Failed reps before halting**. Failed repetitions typically result from random errors driving the model into a region in which it is not defined, for example where the model is forced to take the log or square root of a negative number. When a repetition fails, EViews will discard any partial results from that repetition, then check whether the total number of failures exceeds the threshold set in the **% Failed reps before halting** box. The simulation continues until either this threshold is exceeded, or the target number of successful repetitions is met.

Note, however, that even one failed repetition indicates that care should be taken when interpreting the simulation results, since it indicates that the model is ill-defined for some possible draws of the random components. Simply discarding these extreme values may create misleading results, particularly when the tails of the distribution are used to measure the error bounds of the system.

The **Confidence interval** box sets options for how confidence intervals should be calculated, assuming they have been selected. The **Calc from entire sample** option uses the sample quantile as an estimate of the quantile of the underlying distribution. This involves storing complete tails for the observed outcomes. This can be very memory intensive since the memory used increases linearly in the number of repetitions. The **Reduced memory approx** option uses an updating algorithm due to Jain and Chlamtac (1985). This requires much less memory overall, and the amount used is independent of the number of repetitions. The updating algorithm should provide a reasonable estimate of the tails of the underlying distribution as long as the number of repetitions is not too small.

The **Interval size (2 sided)** box lets you select the size of the confidence interval given by the upper and lower bounds. The default size of 0.95 provides a 95% confidence interval with a weight of 2.5% in each tail. If, instead, you would like to calculate the interquartile range for the simulation results, you should input 0.5 to obtain a confidence interval with bounds at the 25% and 75% quantiles.

The **Innovation covariance** box on the right side of the dialog determines how the innovations to stochastic equations will be generated. At each observation of a stochastic simulation, a set of independent random numbers are drawn from the standard normal distribution, then these numbers are scaled to match the desired variance-covariance matrix of the system. In the general case, this involves multiplying the vector of random numbers by the Cholesky factor of the covariance matrix. If the matrix is diagonal, this reduces to multiplying each random number by its desired standard deviation.

The **Scale variances to match equation specified standard deviations** box lets you determine how the variances of the residuals in the equations are determined. If the box is not checked, the variances are calculated from the model equation residuals. If the box is checked, then any equation that contains a specified standard deviation will use that number instead (see page 797 for details on how to specify a standard deviation from the equation properties page). Note that the sample used for estimation in a linked equation may differ from the sample used when estimating the variances of the model residuals.

The **Diagonal covariance matrix** box lets you determine how the off diagonal elements of the covariance matrix are determined. If the box is checked, the off diagonal elements are set to zero. If the box is not checked, the off diagonal elements are set so that the correlation of the random draws matches the correlation of the observed equation residuals. If the variances are being scaled, this will involve rescaling the estimated covariances so that the correlations are maintained.

The **Estimation sample** box allows you to specify the set of observations that will be used when estimating the variance-covariance matrix of the model residuals. By default, EViews will use the default workfile sample.

The **Multiply covariance matrix** field allows you to set an overall scale factor to be applied to the entire covariance matrix. This can be useful for seeing how the stochastic behavior of the model changes as levels of random variation are applied which are different from those that were observed historically, or as a means of trouble-shooting the model by reducing the overall level of random variation if the model behaves badly.

As noted above, stochastic simulation may include both coefficient uncertainty and exogenous variable uncertainty. There are very different ways methods of specifying these two types of uncertainty.

The **Include coefficient uncertainty** field at the bottom right of the **Stochastic Options** dialog specifies whether estimated coefficients in linked equations should be varied randomly during a stochastic simulation. When this option is selected, coefficients are randomly redrawn at the beginning of each repetition, using the coefficient variability in the estimated equation, if possible. This technique provides a method of incorporating uncertainty surrounding the true values of the coefficients into variation in our forecast results. Note that coefficient uncertainty is ignored in nonlinear equations and in linear equations estimated with PDL terms.

We emphasize that the dynamic behavior of a model may be altered considerably when the coefficients in the model are varied randomly. A model which is stable may become unstable, or a model which converges exponentially may develop cyclical oscillations. One consequence is that the standard errors from a stochastic simulation of a single equation may vary from the standard errors obtained when the same equation is forecast using the EViews equation object. This result arises since the equation object uses an analytic approach to calculating standard errors based on a local linear approximation that effectively imposes stationarity on the original equation.

To specify exogenous variable uncertainty, you must provide information about the variability of each relevant exogenous variable. First, display the model in *Variable View* by selecting **View/Variables** or clicking on the **Variables** button in the toolbar. Next, select the exogenous variable in question, and right mouse click, select **Properties...**, and enter the exogenous variable variance in the resulting dialog. If you supply a positive value, EViews will incorporate exogenous variable uncertainty in the simulation; if the variance is not a valid value (negative or NA), the exogenous variable will be treated as deterministic.

Tracked Variables

The Tracked Variables page of the dialog lets you examine and modify which endogenous variables are being tracked by the model. When a variable is tracked, the results for that variable are saved in a series in the workfile after the simulation is complete. No results are saved for variables that are not tracked.

Tracking is most useful when working with large models, where keeping the results for every endogenous variable in the model would clutter the workfile and use up too much memory.

By default, all variables are tracked. You can switch on selective tracking using the radio button at the top of the dialog. Once selective tracking is selected, you can type in variable names in the dialog below, or use the properties dialog for the endogenous variable to switch tracking on and off.

You can also see which variables are currently being tracked using the variable view, since the names of tracked variables appear in blue.

Diagnostics

The **Diagnostics** dialog page lets you set options to control the display of intermediate output. This can be useful if you are having problems getting your model to solve.

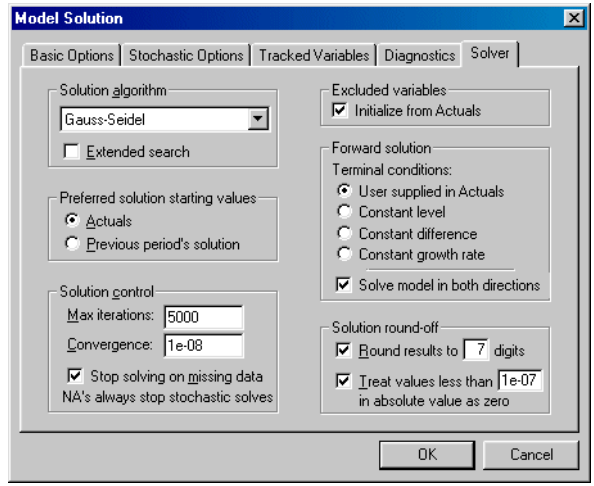
When the **Display detailed messages** box is checked, extra output will be produced in the solution messages window as the model is solved.

The traced variables list lets you specify a list of variables for which intermediate values will be stored during the iterations of the solution process. These results can be examined by switching to the **Trace Output** view after the model is complete. Tracing intermediate values may give you some idea of where to look for problems when a model is generating errors or failing to converge.

Solver

The **Solver** dialog page sets options relating to the non-linear equation solver which is applied to the model.

The **Solution algorithm** box lets you select the algorithm that will be used to solve the model for a single period. The following choices are available:



- **Gauss-Seidel**: the Gauss-Seidel algorithm is an iterative algorithm, where at each iteration we solve each equation in the model for the value of its associated endogenous variable, treating all other endogenous variables as fixed. This algorithm requires little working memory and has fairly low computational costs, but requires the equation system to have certain stability properties for it to converge. Although it is easy to construct models that do not satisfy these properties, in practice, the algorithm generally performs well on most econometric models. If you are having difficulties with the algorithm, you might like to try reordering the equations, or rewriting the equations to change the assignment of endogenous variables, since these changes can affect the stability of the Gauss-Seidel iterations.

- **Newton**: Newton's method is also an iterative method, where at each iteration we take a linear approximation to the model, then solve the linear system to find a root of the model. This algorithm can handle a wider class of problems than Gauss-Seidel, but requires considerably more working memory and has a much greater computational cost when applied to large models. Newton's method is invariant to equation reordering or rewriting.

Note that even if Newton's method is selected for solving within each period of the model, a Gauss-Seidel type method is used between all the periods if the model requires iterative forward solution. See "Models Containing Future Values" on page 805.

The **Excluded variables/Initialize from Actuals** checkbox controls where EViews takes values for excluded variables. By default, this box is checked and all excluded observations for solved endogenous variables (both in the solution sample and pre-solution observations) are initialized to the actual values of the endogenous variables prior to the start of a model solution. If this box is unchecked, EViews will initialize the excluded variables with values from the solution series (aliased series), so that you may set the values manually without editing the original series.

The **Extended search** checkbox tells the solver to try alternative step sizes when searching for new values for the endogenous variables during an iteration. This improves the chances of convergence, but will generally increase the time taken to solve the model. If your model is having difficulty converging, you may like to try this option.

The **Preferred solution starting values** section lets you select the values to be used as starting values in the iterative procedure. When **Actuals** is selected, EViews will first try to use values contained in the actuals series as starting values. If these are not available, EViews will try to use the values solved for in the previous period. If these are not available, EViews will default to using arbitrary starting values of 0.1. When **Previous period's solution** is selected, the order is changed so that the previous periods values are tried first, and only if they are not available, are the actuals used.

The **Solution contro**l section allows you to set termination options for the solver. **Max iterations** sets the maximum number of iterations that the solver will carry out before aborting. **Convergence** sets the threshold for the convergence test. If the largest relative change between iterations of any endogenous variable has an absolute value less than this threshold, then the solution is considered to have converged. **Stop on missing data** means that the solver should stop as soon as one or more exogenous (or lagged endogenous) variables is not available. If this option is not checked, the solver will proceed to subsequent periods, storing NAs for this period's results.

The **Forward solution** section allows you to adjust options that affect how the model is solved when one or more equations in the model contain future (forward) values of the endogenous variables. The **Terminal conditions** section lets you specify how the values of the endogenous variables are determined for leads that extend past the end of the forecast period. If **User supplied in Actuals** is selected, the values contained in the Actuals series after the end of the forecast sample will be used as fixed terminal values. If no values are available, the solver will be unable to proceed. If **Constant level** is selected, the terminal values are determined endogenously by adding the condition to the model that the values of the endogenous variables are constant over the post-forecast period at the same level as the final forecasted values ($y_t = y_{t-1}$ for $t = T, T+1, \ldots, T+k-1$), where T is the first observation past the end of the forecast sample, and k is the maximum lead in the model). This option may be a good choice if the model converges to a stationary state. If **Constant difference** is selected, the terminal values are determined endogenously by adding the condition that the values of the endogenous variables follow a linear trend over the post forecast period, with a slope given by the difference between the last two forecasted values:

$$y_t - y_{t-1} = y_{t-1} - y_{t-2} \tag{26.8}$$

for $t = T, T+1, \ldots, T+k-1$). This option may be a good choice if the model is in log form and tends to converge to a steady state. If **Constant growth rate** is selected, the terminal values are determined endogenously by adding the condition to the model that the

endogenous variables grow exponentially over the post-forecast period, with the growth rate given by the growth between the final two forecasted values:

$$(y_t - y_{t-1})/y_{t-1} = (y_{t-1} - y_{t-2})/y_{t-2} \tag{26.9}$$

for $t = T, T+1, \ldots, T+k-1$). This latter option may be a good choice if the model tends to produce forecasts for the endogenous variables which converge to constant growth paths.

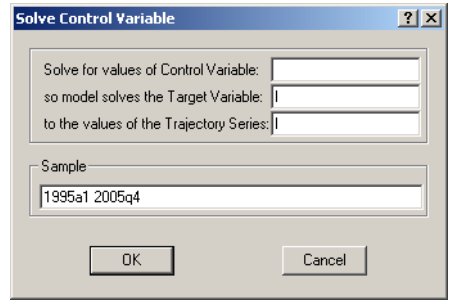
The **Solve in both directions** option affects how the solver loops over periods when calculating forward solutions. When the box is not checked, the solver always proceeds from the beginning to the end of the forecast period during the Gauss-Seidel iterations. When the box is checked, the solver alternates between moving forwards and moving backwards through the forecast period. The two approaches will generally converge at slightly different rates depending on the level of forward or backward persistence in the model. You should choose whichever setting results in a lower iteration count for your particular model.

The **Solution round-off** section of the dialog controls how the results are rounded after convergence has been achieved. Because the solution algorithms are iterative and provide only approximate results to a specified tolerance, small variations can occur when comparing solutions from models, even when the results should be identical in theory. Rounding can be used to remove some of this minor variation so that results will be more consistent. The default settings will normally be adequate, but if your model has one or more endogenous variables of very small magnitude, you will need to switch off the rounding to zero or rescale the variables so that their solutions are farther from zero.

Solve Control for Target

Normally, when solving a model, we start with a set of known values for our exogenous variables, then solve for the unknown values of the endogenous variables of the model. If we would like an endogenous variable in our model to follow a particular path, we can solve the model repeatedly for different values of the exogenous variables, changing the values until the path we want for the endogenous variable is produced. For example, in a macroeconomic model, we may be interested in examining what value of the personal tax rate would be needed in each period to produce a balanced budget over the forecast horizon.

The problem with carrying out this procedure by hand is that the interactions between variables in the model make it difficult to guess the correct values for the exogenous variables. It will often require many attempts to find the values that solve the model to give the desired results.



To make this process easier, EViews provides a special procedure for solving a model which automatically searches for the unknown values. Simply create a series in the workfile which contains the values you would like the endogenous variable to achieve, then select **Proc/Solve Control for Target…** from the menus. Enter the name of the exogenous variable you would like to modify in the **Control Variable** box, the name of the endogenous variable which you are targeting in the **Target Variable** box, and the name of the workfile series which contains the target values in the **Trajectory Variable** box. Set the sample to the range for you would like to solve, then click on **OK**.

The procedure may take some time to complete, since it involves repeatedly solving the model to search for the desired solution. It is also possible for the procedure to fail if it cannot find a value of the exogenous variable for which the endogenous variable solves to the target value. If the procedure fails, you may like to try moving the trajectory series closer to values that you are sure the model can achieve.

Working with the Model Data

When working with a model, much of your time will be spent viewing and modifying the data associated with the model. Before solving the model, you will edit the paths of your exogenous variables or add factors during the forecast period. After solving the model, you will use graphs or tables of the endogenous variables to evaluate the results. Because there is a large amount of data associated with a model, you will also spend time simply managing the data.

Since all the data associated with a model is stored inside standard series in the workfile, you can use all of the usual tools in EViews to work with the data of your model. However, it is often more convenient to work directly from the model window.

Although there are some differences in details, working with the model data generally involves following the same basic steps. You will typically first use the variable view to select the set of variables you would like to work with, then use either the right mouse button menu or the model procedure menu to select the operation to perform.

Because there may be several series in the workfile associated with each variable in the model, you will then need to select the types of series with which you wish to work. The following types will generally be available:

- Actuals: the workfile series with the same name as the variable name. This will typically hold the historical data for the endogenous variables, and the historical data and baseline forecast for the exogenous variables.
- Active: the workfile series that is used when solving the active scenario. For endogenous variables, this will be the series with a name consisting of the variable name followed by the scenario extension. For exogenous variables, the actual series will be used unless it has been overridden. In this case, the exogenous variable will also be the workfile series formed by appending the scenario extension to the variable name.
- Alternate: the workfile series that is used when solving the alternate scenario. The rules are the same as for active.

In the following sections, we discuss how different operations can be performed on the model data from within the variable view.

Editing Data

The easiest way to make simple changes to the data associated with a model is to open a series or group spreadsheet window containing the data, then edit the data by hand.

To open a series window from within the model, simply select the variable using the mouse in the variable view, then use the right mouse button menu to choose **Open selected series...**, followed by **Actuals**, **Active Scenario** or **Alternate Scenario**. If you select several series before using the option, an unnamed group object will be created to hold all the series.

To edit the data, click the **Edit +/-** button to make sure the spreadsheet is in edit mode. You can either edit the data directly in levels or use the **Units** button to work with a transformed form of the data, such as the differences or percentage changes.

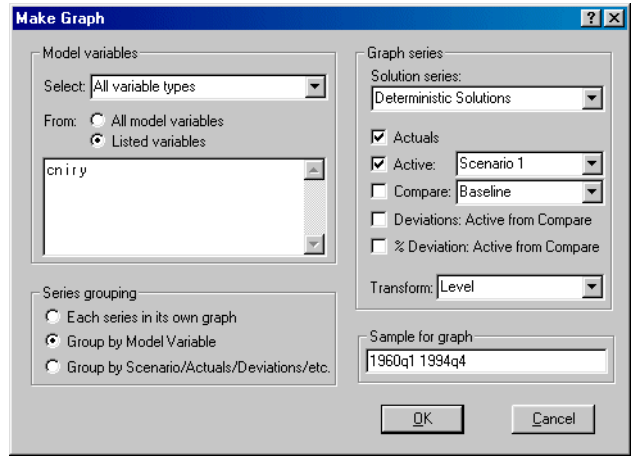
To create a group which allows you to edit more than one of the series associated with a variable at the same time, you can use the **Make Group/Table** procedure discussed below to create a dated data table, then switch the group to spreadsheet view to edit the data.

More complicated changes to the data may require using a genr command to calculate the series by specifying an expression. Click the **Genr** button from the series window toolbar to call up the dialog, then type in the expression to generate values for the series and set the workfile sample to the range of values you would like to modify.

Displaying Data

The EViews model object provides two main forms in which to display data: as a graph or as a table. Both of these can be generated easily from the model window.

From the variable view, select the variables you wish to display, then use the right mouse button menu or the main menu to select **Proc** and then **Make Group/Table** or **Make Graph**.



The dialogs for the two procs are almost identical. Here we see the **Make Graph** dialog. We saw this dialog earlier in our macro model example. The majority of fields in the dialog control which series you would like the table or graph to contain. At the top left of the graph is the **Model Variables** box, which is used to select the set of variables to place in the graph. By default, the table or graph will contain the variables that are currently selected in the variable view. You can expand this to include all model variables, or add or remove particular variables from the list of selected variables using the radio buttons and text box labeled **From**. You can also restrict the set of variables chosen according to variable type using the list box next to **Select**. By combining these fields, it is easy to select sets of variables such as all of the endogenous variables of the model, or all of the overridden variables.

Once the set of variables has been determined, it is necessary to map the variable names into the names of series in the workfile. This typically involves adding an extension to each name according to which scenario the data is from and the type of data contained in the series. The options affecting this are contained in the **Graph series** (if you are making a graph) or **Series types** (if you are making a group/table) box at the right of the dialog.

The **Solution series** box lets you choose which solution results you would like to examine when working with endogenous variables. You can choose from a variety of series generated during deterministic or stochastic simulations.

The series of checkboxes below determine which scenarios you would like to display in the graphs, as well as whether you would like to calculate deviations between various scenarios. You can choose to display the actual series, the series from the active scenario, or the series from an alternate scenario (labeled “Compare”). You can also display either the difference between the active and alternate scenario (labeled “Deviations: Active from Compare”), or the ratio between the active and alternate scenario in percentage terms (labeled “% Deviation: Active from Compare”).

The final field in the **Graph series** or **Series types** box is the **Transform** listbox. This lets you apply a transformation to the data similar to the **Transform** button in the series spreadsheet.

While the deviations and units options allow you to present a variety of transformations of your data, in some cases you may be interested in other transformations that are not directly available. Similarly, in a stochastic simulation, you may be interested in examining standard errors or confidence bounds on the transformed series, which will not be available when you apply transformations to the data after the simulation is complete. In either of these cases, it may be worth adding an identity to the model that generates the series you are interested in examining as part of the model solution.

For example, if your model contains a variable GDP, you may like to add a new equation to the model to calculate the percentage change of GDP:

```
pgdp = @pch(gdp)
```

After you have solved the model, you can use the variable PGDP to examine the percentage change in GDP, including examining the error bounds from a stochastic simulation. Note that the cost of adding such identities is relatively low, since EViews will place all such identities in a final recursive block which is evaluated only once after the main endogenous variables have already been solved.

The remaining option, at the bottom left of the dialog, lets you determine how the series will be grouped in the output. The options are slightly different for tables and graphs. For tables, you can choose to either place all series associated with the same model variable together, or to place each series of the same series type together. For graphs, you have the same two choices, and one additional choice, which is to place every series in its own graph.

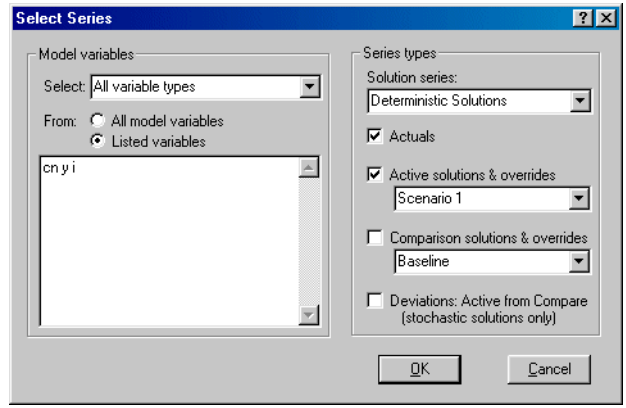
In the graph dialog, you also have the option of setting a sample for the graph. This is often useful when you are plotting forecast results since it allows you to choose the amount of historical data to display in the graph prior to the forecast results. By default, the sample is set to the workfile sample.

When you have finished setting the options, simply click on **OK** to create the new table or graph. All of EViews usual editing features are available to modify the table or graph for final presentation.

Managing Data

When working with a model, you will often create many series in the workfile for each variable, each containing different types of results or the data from different scenarios. The model object provides a number of tools to help you manage these series, allowing you to perform copy, fetch, store and delete operations directly from within the model.

Because the series names are related to the variable names in a consistent way, management tasks can often also be performed from outside the model by using the pattern matching features available in EViews commands (see Appendix B, "Wildcards", on page 945).



The data management operations from within the model window proceed very similarly to the data display operations. First, select the variables you would like to work with from the variable view, then choose **Copy**, **Store series…**, **Fetch series…** or **Delete series…** from the right mouse button menu or the object procedures menu. A dialog will appear, similar to the one used when making a table or graph.

In the same way as for the table and graph dialogs, the left side of the dialog is used to choose which of the model variables to work with, while the right side of the dialog is used to select one or more series associated with each variable. Most of the choices are exactly the same as for graphs and tables. One significant difference is that the checkboxes for active and comparison scenarios include exogenous variables only if they have been overridden in the scenario. Unlike when displaying or editing the data, if an exogenous variable has not been overridden, the actual series will not be included in its place. The only way to store, fetch or delete any actual series is to use the **Actuals** checkbox.

After clicking on **OK**, you will receive the usual prompts for the store, fetch and delete operations. You can proceed as usual.

Part VI. Panel and Pooled Data

Panel and pool data involve observations that possess both cross-section, and within-cross-section identifiers.

Generally speaking, we distinguish between the two by noting that pooled time-series, cross-section data refer to data with relatively few cross-sections, where variables are held in cross-section specific individual series, while panel data correspond to data with large numbers of cross-sections, with variables held in single series in stacked form.

The discussion of these data is divided into parts. Pooled data structures are discussed first:

- Chapter 27, "Pooled Time Series, Cross-Section Data", on page 825 outlines tools for working with pooled time series, cross-section data, and estimating standard equation specifications which account for the pooled structure of the data.

Panel data are described separately. In Chapter 9, "Advanced Workfiles", beginning on page 207, we describe the basics of structuring a workfile for use with panel data. Once a workfile is structured as a panel workfile, EViews provides you with different tools for working with data in the workfile, and for estimating equation specifications using both the data and the panel structure.

- Chapter 28, "Working with Panel Data", beginning on page 873, outlines the basics of working with panel workfiles.
- Chapter 29, "Panel Estimation", beginning on page 901 describes estimation in panel structured workfiles.

Chapter 27. Pooled Time Series, Cross-Section Data

Data often contain information on a relatively small number of cross-sectional units observed over time. For example, you may have time series data on GDP for a number of European nations. Or perhaps you have state level data on unemployment observed over time. We term such data *pooled* time series, cross-section data.

EViews provides a number of specialized tools to help you work with pooled data. EViews will help you manage your data, perform operations in either the time series or the cross-section dimension, and apply estimation methods that account for the pooled structure of your data.

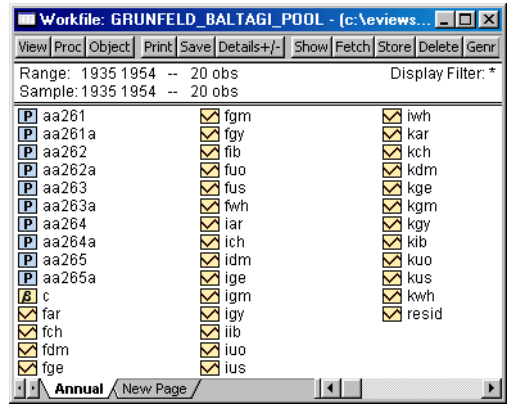
The EViews object that manages time series/cross-section data is called a *pool*. The remainder of this chapter will describe how to set up your data to work with pools, and how to define and work with pool objects.

Note that the data structures described in this chapter should be distinguished from data where there are large numbers of cross-sectional units. This type of data is typically termed *panel* data. Working with panel structured data in EViews is described in Chapter 28, "Working with Panel Data", on page 873 and Chapter 29, "Panel Estimation", beginning on page 901.

The Pool Workfile

The first step in working with pooled data is to set up a *pool workfile*. There are several characteristics of an EViews workfile that allow it to be used with pooled time series, cross-section data.

First, a pool workfile is an ordinary EViews workfile structured to match the time series dimension of your data. The range of your workfile should represent the earliest and latest dates or observations you wish to consider for any of the cross-section units. For example, if you want to work with data for some firms from 1932 to 1954, and data for other firms from 1930 to 1950, you should create a workfile ranging from 1930 to 1954.


Workfile: GRUNFELD_BALTAGI_POOL - (c:\eviews...
View Proc Object Print Save Details+/- Show Fetch Store Delete Genr
Range: 1935 1954 -- 20 obs
Sample: 1935 1954 -- 20 obs
Display Filter: *
aa261 aa261a aa262 aa262a aa263 aa263a aa264 aa264a aa265 aa265a c far fch fdm fge
fgm fgy fib fuo fus fwh iar ich idm ige igm igy iib iuo ius
iwh kar kch kdm kge kgm kgy kib kuo kus kwh resid
Annual
New Page

Second, the pool workfile should contain EViews series that follow a user-defined naming convention. For each cross-section

specific variable, you should have a separate series corresponding to each cross-section/variable combination. For example, if you have time series data for an economic variable like investment that differs for each of 10 firms, you should have 10 separate investment series in the workfile with names that follow the user-defined convention.

Lastly, and most importantly, a pool workfile must contain one or more *pool objects*, each of which contains a (possibly different) description of the pooled structure of your workfile in the form of rules specifying the user-defined naming convention for your series.

There are various approaches that you may use to set up your pool workfile:

- First, you may simply create a new workfile in the usual manner, by describing, the time series structure of your data. Once you have a workfile with the desired structure, you may define a pool object, and use this object as a tool in creating the series of interest and importing data into the series.
- Second, you may create an EViews workfile containing your data in stacked form. Once you have your stacked data, you may use the built-in workfile reshaping tools to create a workfile containing the desired structure and series.

Both of these procedures require a bit more background on the nature of the pool object, and the way that your pooled data are held in the workfile. We begin with a brief description of the basic components of the pool object, and then return to a description of the task of setting up your workfile and data ("Setting up a Pool Workfile" on page 831).

The Pool Object

Before describing the pooled workfile in greater detail, we must first provide a brief description of the EViews *pool object*.

We begin by noting that the pool object serves two distinct roles. First, the pool contains a set of definitions that describe the structure of the pooled time series, cross-section data in your workfile. In this role, the pool object serves as a tool for managing and working with pooled data, much like the group object serves is used as a tool for working with sets of series. Second, the pool provides procedures for estimating econometric models using pooled data, and examining and working with the results from this estimation. In this role, the pool object is analogous to an equation object that is used to estimate econometric specifications.

In this section, we focus on the definitions that serve as the foundation for the pool object and simple tools for managing your pool object. The tools for working with data are described in "Working with Pooled Data" beginning on page 838, and the role of the pool object in estimation is the focus of "Pooled Estimation" beginning on page 845.

Defining a Pool Object

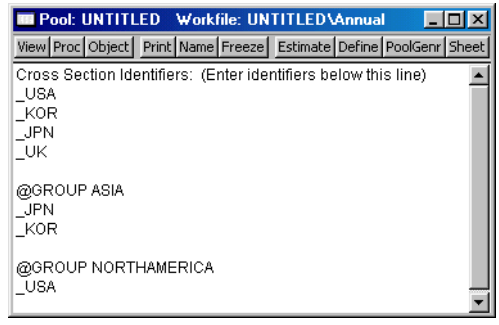
There are two parts to the definitions in a pool object: the cross-section identifiers, and optionally, definitions of groups of identifiers.

Cross-section Identifiers

The central feature of a pool object is a list of cross-section members which provides a naming convention for series in the workfile. The entries in this list are termed *cross-section identifiers*. For example, in a cross-country study, you might use "_USA" to refer to the United States, "_KOR" to identify Korea, "_JPN" for Japan, and "_UK" for the United Kingdom. Since the cross-section identifiers will be used as a base in forming series names, we recommend that they be kept relatively short.

Specifying the list cross-section identifiers in a pool tells EViews about the structure of your data. When using a pool with the four cross-section identifiers given above, you instruct EViews to work with separate time series data for each of the four countries, and that the data may be held in series that contain the identifiers as part of the series names.

The most direct way of creating a pool object is to select **Object/New Object.../Pool...**. EViews will open the pool specification view into which you should enter or copy-and-paste a list of identifiers, with individual entries separated by spaces, tabs, or carriage returns. Here, we have entered four identifiers on separate lines.



There are no special restrictions on the labels that you can use for cross-section identifiers, though you must be able to form legal EViews series names containing these identifiers.

Note that we have used the "_" character at the start of each of the identifiers in our list; this is not necessary, but you may find that it makes it easier to spot the identifier when it is used as the end of a series name.

Before moving on, it is important to note that a pool object is simply a description of the underlying structure of your data, so that it does not itself contain series or data. This separation of the object and the data has important consequences.

First, you may use pool objects to define multiple sets of cross-section identifiers. Suppose, for example, that the pool object POOL01 contains the definitions given above. You may also have a POOL02 that contains the identifiers "_GER", "_AUS", "_SWTZ", and a POOL03 that contains the identifiers "_JPN" and "_KOR". Each of these three pool objects

defines a different set of identifiers, and may be used to work with different sets of series in the workfile. Alternatively, you may have multiple pool objects in a workfile, each of which contain the same list of identifiers. A POOL04 that contains the same identifiers as POOL01 may be used to work with data from the same set of countries.

Second, since pool objects contain only definitions and not series data, deleting a pool will not delete underlying series data. You may, however, *use* a pool object to delete, create, and manipulate underlying series data.

Group Definitions

In addition to the main list of cross-section identifiers, you may define groups made up of subsets of your identifiers. To define a group of identifiers, you should enter the keyword "@GROUP" followed by a name for the group, and the subset of the pool identifiers that are to be used in the group. EViews will define a group using the specified name and any identifiers provided.

We may, for example, define the ASIA group containing the "_JPN" and "_KOR" identifiers, or the NORTHAMERICA group containing the "_USA" identifier by adding:

```
@group asia _jpn _kor
@group northamerica _usa
```

to the pool definition.

These subsets of cross-section identifiers may be used to define virtual series indicating whether a given observation corresponds to a given subgroup or not. The ASIA group, for example, can be used along with special tools to identify whether a given observation should be viewed as coming from Japan or Korea, or from one of the other countries in the pool. We describe this functionality in greater detail in "Pool Series" on page 830.

Viewing or Editing Definitions

You may, at any time, change the view of an existing pool object to examine the current list of cross-section identifiers and group definitions. Simply push the **Define** button on the toolbar, or select **View/Cross-Section Identifiers**. If desired, you can edit the list of identifiers or group definitions.

Copying a Pool Object

Typically, you will work with more than one pool object. Multiple pools are used to define various subsamples of cross-section identifiers, or to work with different pooled estimation specifications.

To copy a pool object, open the original pool, and select **Object/Copy Object…** Alternatively, you can highlight the name of the pool in the workfile window, select **Object/Copy Selected…** and enter the new name.

Pooled Data

As noted previously, all of your pooled data will be held in ordinary EViews series. These series can be used in all of the usual ways: they may, among other things, be tabulated, graphed, used to generate new series, or used in estimation. You may also use a pool object to work with sets of the individual series.

There are two classes of series in a pooled workfile: *ordinary series* and *cross-section specific series*.

Ordinary Series

An ordinary series is one that has common values across all cross-sections. A single series may be used to hold the data for each variable, and these data may be applied to every cross-section. For example, in a pooled workfile with firm cross-section identifiers, data on overall economic conditions such as GDP or money supply do not vary across firms. You need only create a single series to hold the GDP data, and a single series to hold the money supply variable.

Since ordinary series do not interact with cross-sections, they may be defined without reference to a pool object. Most importantly, there are no naming conventions associated with ordinary series beyond those for ordinary EViews objects.

Cross-section Specific Series

Cross-section specific series are those that have values that differ between cross-sections. A set of these series are required to hold the data for a given variable, with each series corresponding to data for a specific cross-section.

Since cross-section specific series interact with cross-sections, they should be defined in conjunction with the identifiers in pool objects. Suppose, for example, that you have a pool object that contains the identifiers "_USA", "_JPN", "_KOR" and "_UK", and that you have time series data on GDP for each of the cross-section units. In this setting, you should have a four cross-section specific GDP series in your workfile.

The key to naming your cross-section specific series is to use names that are a combination of a *base name* and a cross-section identifier. The cross-section identifiers may be embedded at an arbitrary location in the series name, so long as this is done consistently across identifiers.

You may elect to place the identifier at the end of the base name, in which case, you should name your series "GDP_USA", "GDP_JPN", "GDP_KOR", and "GDP_UK". Alternatively, you may choose to put the section identifiers in front of the name, so that you have the names "_USAGDP", "_JPNGDP", "_KORGDP", and "_UKGDP". The identifiers may

also be placed in the middle of series names—for example, using the names "GDP_USAINF" "GDP_JPNIN", "GDP_KORIN", "GDP_UKIN".

It really doesn't matter whether the identifiers are used at the beginning, middle, or end of your cross-section specific names; you should adopt a naming style that you find easiest to manage. Consistency in the naming of the set of cross-section series is, however, absolutely essential. You should not, for example, name your four GDP series "GDP_USA", "_JPNGDPIN", "GDP_KOR", "_UKGDP", as this will make it impossible for EViews to refer to the set of series using a pool object.

Pool Series

Once your series names have been chosen to correspond with the identifiers in your pool, the pool object can be used to work with a set of series as though it were a single item. The key to this processing is the concept of a *pool series*.

A pool series is actually a set of series defined by a base name and the entire list of cross-section identifiers in a specified pool. Pool series are specified using the base name, and a "?" character placeholder for the cross-section identifier. If your series are named "GDP_USA", "GDP_JPN", "GDP_KOR", and "GDP_UK", the corresponding pool series may be referred to as "GDP?". If the names of your series are "_USAGDP", "_JPNGDP", "_KORGDP", and "_UKGDP", the pool series is "?GDP".

When you use a pool series name, EViews understands that you wish to work with all of the series in the workfile that match the pool series specification. EViews loops through the list of cross-section identifiers in the specified pool, and substitutes each identifier in place of the "?". EViews then uses the complete set of cross-section specific series formed in this fashion.

In addition to pool series defined with "?", EViews provides a special function, @INGRP, that you may use to generate a group identity pool series that takes the value 1 if an observation is in the specified group, and 0 otherwise.

Consider, for example, the @GROUP for "ASIA" defined using the identifiers "_JPN" and "_KOR", and suppose that we wish to create a dummy variable series for whether an observation is in the group. One approach to representing these data is to create the following four cross-section specific series:

```
series asia_jpn = 1
series asia_kor = 1
series asia_usa = 0
series asia_uk = 0
```

and to refer to them collectively as the pool series "ASIA_?". While not particularly difficult to do, this direct approach becomes more cumbersome the greater the number of cross-section identifiers.

More easily, we may use the special pool series expression:

```
@ingrp(asia)
```

to define a special virtual pool series in which each observation takes a 0 or 1 indicator for whether an observation is in the specified group. This expression is equivalent to creating the four cross-section specific series, and referring to them as "ASIA_?".

We must emphasize that pool series specifiers using the "?" and the @INGRP function may only be used through a pool object, since they have no meaning without a list of cross-section identifiers. If you attempt to use a pool series outside the context of a pool object, EViews will attempt to interpret the "?" as a wildcard character (see Appendix B, "Wildcards", on page 945). The result, most often, will be an error message saying that your variable is not defined.

Setting up a Pool Workfile

Your goal in setting up a pool workfile is to obtain a workfile containing individual series for ordinary variables, sets of appropriately named series for the cross-section specific data, and pool objects containing the related sets of identifiers. The workfile should have frequency and range matching the time series dimension of your pooled data.

There are two basic approaches to setting up such a workfile. The direct approach involves first creating an empty workfile with the desired structure, and then importing data into individual series using either standard or pool specific import methods. The indirect approach involves first creating a stacked representation of the data in EViews, and then using EViews built-in reshaping tools to set up a pooled workfile.

Direct Setup

The direct approach to setting up your pool workfile involves three distinct steps: first creating a workfile with the desired time series structure; next, creating one or more pool objects containing the desired cross-section identifiers; and lastly, using pool object tools to import data into individual series in the workfile.

Creating the Workfile and Pool Object

The first step in the direct setup is to create an ordinary EViews workfile structured to match the time series dimension of your data. The range of your workfile should represent the earliest and latest dates or observations you wish to consider for any of the cross-section units.

Simply select **File/New workfile...** to bring up the **Workfile Create** dialog which you will use to describe the structure of your workfile. For additional detail, see "Creating a Workfile by Describing its Structure" on page 51.

For example, to create a pool workfile that has annual data ranging from 1950 to 1992, simply select **Annual** in the **Frequency** combo box, and enter "1950" as the **Start date** and "1992" as the **End date**.

Next, you should create one or more pool objects containing cross-section identifiers and group definitions as described in "The Pool Object" on page 826.

Importing Pooled Data

Lastly, you should use one of the various methods for importing data into series in the workfile. Before considering the various approaches, we require an understanding the various representations of pooled time series, cross-section data that you may encounter.

Bear in mind that in a pooled setting, a given observation on a variable may be indexed along three dimensions: the variable, the cross-section, and the time period. For example, you may be interested in the value of GDP, for the U.K., in 1989.

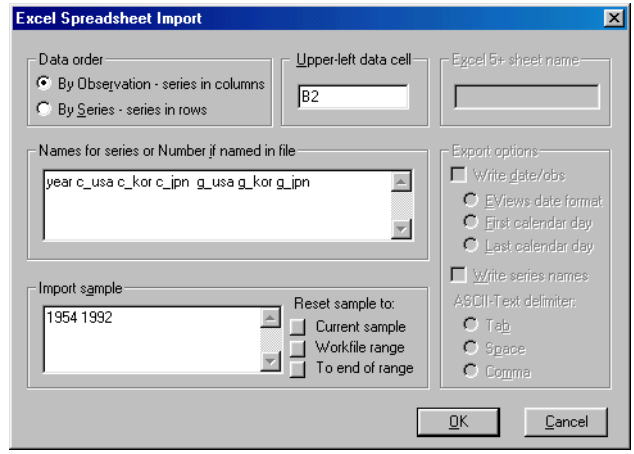
Despite the fact that there are three dimensions of interest, you will eventually find yourself working with a two-dimensional representation of your pooled data. There is obviously no unique way to organize three-dimensional data in two-dimensions, but several formats are commonly employed.

Unstacked Data

In this form, observations on a given variable for a given cross-section are grouped together, but are separated from observations for other variables and other cross sections. For example, suppose the top of our Excel data file contains the following:

year	c_usa	c_kor	c_jpn	g_usa	g_jpn	g_kor
1954	61.6	77.4	66	17.8	18.7	17.6
1955	61.1	79.2	65.7	15.8	17.1	16.9
1956	61.7	80.2	66.1	15.7	15.9	17.5
1957	62.4	78.6	65.5	16.3	14.8	16.3
…	…	…	…	…	…	…

Here, the base name "C" represents consumption, while "G" represents government expenditure. Each country has its own separately identified column for consumption, and its own column for government expenditure.



EViews pooled workfiles are structured to work naturally with data that are unstacked, since the sets of cross-section specific series in the pool workfile correspond directly to the multiple columns of unstacked source data. You may read unstacked data directly into EViews using the standard import procedures described in "Frequency Conversion" on page 115. Simply read each cross-section specific variable as an individual series, making certain that the names of the resulting series follow the pool naming conventions given in your pool object. Ordinary series may be imported in the usual fashion with no additional complications.

In this example, we use the standard EViews import tools to read separate series for each column. We create the individual series "YEAR", "C_USA", "C_KOR", "C_JPN", "G_USA", "G_JPN", and "G_KOR".

Stacked Data

Pooled data can also be arranged in stacked form, where all of the data for a variable are grouped together in a single column.

In the most common form, the data for different cross-sections are stacked on top of one another, with all of the sequentially dated observations for a given cross-section grouped together. We may say that these data are *stacked by cross-section*:

id	year	c	g
_usa	1954	61.6	17.8
_usa	…	…	…
_usa	…	…	…
_usa	1992	68.1	13.2
…	…	…	…
_kor	1954	77.4	17.6
_kor	…	…	…
_kor	1992	na	na

Alternatively, we may have data that are *stacked by date*, with all of the observations of a given period grouped together:

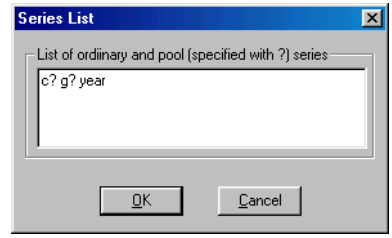
per	id	c	g
1954	_usa	61.6	17.8
1954	_uk	62.4	23.8
1954	_jpn	66	18.7
1954	_kor	77.4	17.6
…	…	…	…
1992	_usa	68.1	13.2
1992	_uk	67.9	17.3
1992	_jpn	54.2	7.6
1992	_kor	na	na

Each column again represents a single variable, but within each column, all of the cross-sections for a given year are grouped together. If data are stacked by year, you should make certain that the ordering of the cross-sectional identifiers within a year is consistent across years.

One straightforward method of importing data into your pool series is by manually entering into, or copying-and-pasting from and into, a stacked representation of your data. First, using the pool object, we will create the stacked representation of the data in EViews:

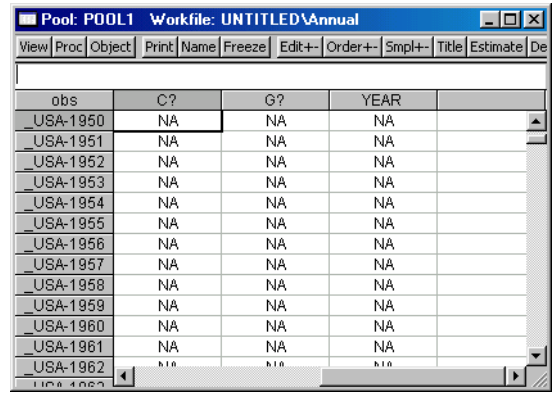
- First, specify which time series observations will be included in your stacked spreadsheet by setting the workfile sample.
- Next, open the pool, then select **View/Spreadsheet View…** EViews will prompt you for a list of series. You can enter ordinary series names or pool series names. If the series exist, then EViews will display the data in the series. If the series do not exist, then EViews will create the series or group of series, using the cross-section identifiers if you specify a pool series.
- EViews will open the stacked spreadsheet view of the pool series. If desired, click on the **Order** +/– button to toggle between stacking by cross-section and stacking by date.
- Click **Edit** +/– to turn on edit mode in the spreadsheet window, and enter your data, or cut-and-paste from another application.

For example, if we have a pool object that contains the identifiers "_USA", "_UK", "_JPN", and "_KOR", we can instruct EViews to create the series C_USA, C_UK, C_JPN, C_KOR, and G_USA, G_UK, G_JPN, G_KOR, and YEAR simply by entering the pool series names "C?", "G?" and the ordinary series name "YEAR", and pressing **OK**.



EViews will open a stacked spreadsheet view of the series in your list. Here we see the series stacked by cross-section, with the pool or ordinary series names in the column header, and the cross-section/date identifiers labeling each row. Note that since YEAR is an ordinary series, its values are repeated for each cross-section in the stacked spreadsheet.

If desired, click on **Order** +/– to toggle between stacking methods to match the organization of the data to be imported. Click on **Edit** +/– to turn on edit mode, and enter or cut-and-paste into the window.



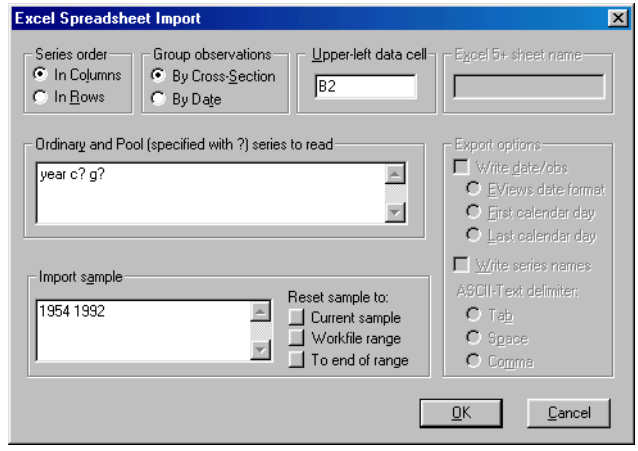
obs	C?	G?	YEAR
_USA-1950	NA	NA	NA
_USA-1951	NA	NA	NA
_USA-1952	NA	NA	NA
_USA-1953	NA	NA	NA
_USA-1954	NA	NA	NA
_USA-1955	NA	NA	NA
_USA-1956	NA	NA	NA
_USA-1957	NA	NA	NA
_USA-1958	NA	NA	NA
_USA-1959	NA	NA	NA
_USA-1960	NA	NA	NA
_USA-1961	NA	NA	NA
_USA-1962	NA	NA	NA

Alternatively, you can import stacked data from a file using import tools built into the pool object. While the data in the file may be stacked either by cross-section or by period, EViews does require that the stacked data are "balanced", and that the cross-sections ordering in the file matches the cross-sectional identifiers in the pool. By "balanced", we mean that if the data are stacked by cross-section, each cross-section should contain exactly the same number of periods—if the data are stacked by date, each date should have exactly the same number of cross-sectional observations arranged in the same order.

We emphasize that only the representation of the data in the import file needs to be balanced; *the underlying data need not be balanced*. Notably, if you have missing values for some observations, you should make certain that there are lines in the file representing the missing values. In the two examples above, the underlying data are not balanced, since information is not available for Korea in 1992. The data in the file have been balanced by including an observation for the missing data.

To import stacked pool data from a file, first open the pool object, then select **Proc/Import Pool data (ASCII, .XLS, .WK?)…**It is important that you use the import procedure associated with the pool object, and not the standard file import procedure.

Select your input file in the usual fashion. If you select a spreadsheet file, EViews will open a spreadsheet import dialog prompting you for additional input.



Much of this dialog should be familiar from the discussion in Chapter 5, "Basic Data Handling", on page 87.

First, indicate whether the pool series are in rows or in columns, and whether the data are stacked by cross-section, or stacked by date.

Next, in the pool series edit box, enter the names of the series you wish to import. This list may contain any combination of ordinary series names and pool series names.

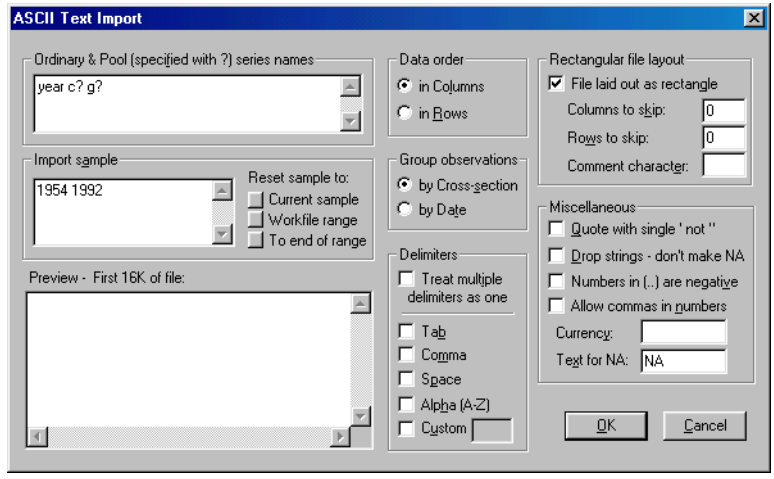
Lastly, fill in the sample information, starting cell location, and optionally, the sheet name.

When you specify your series using pool series names, EViews will, if necessary, create and name the corresponding set of pool series using the list of cross-section identifiers in the pool object. If you list an ordinary series name, EViews will, if needed, create a single series to hold the data.

EViews will read the contents of your file into the specified pool variables using the sample information. When reading into pool series, the first set of observations in the file will be placed in the individual series corresponding to the first cross-section (if reading data that is grouped by cross-section), or the first sample observation of each series in the set of cross-sectional series (if reading data that is grouped by date), and so forth.

If you read data into an ordinary series, EViews will continually assign values into the corresponding observation of the single series, so that upon completion of the import procedure, the series will contain the last set of values read from the file.

The basic technique for importing stacked data from ASCII text files is analogous, but the corresponding dialog contains many additional options to handle the complexity of text files.



For a discussion of the text specific settings in the dialog, see "Importing ASCII Text Files" on page 120.

Indirect Setup (Restructuring)

Second, you may create an ordinary EViews workfile containing your data in stacked form, and then use the workfile reshaping tools to create a pool workfile with the desired structure and contents.

The first step in the indirect setup of a pool workfile is to create a workfile containing the contents of your stacked data file. You may manually create the workfile and import the stacked series data, or you may use EViews tools for opening foreign source data directly into a new workfile ("Creating a Workfile by Reading from a Foreign Data Source" on page 53).

Once you have your stacked data in an EViews workfile, you may use the workfile reshaping tools to unstack the data into a pool workfile page. In addition to unstacking the data into multiple series, EViews will create a pool object containing identifiers obtained from patterns in the series names. See "Reshaping a Workfile" beginning on page 241 for a general discussion of reshaping, and "Unstacking a Workfile" on page 244 for a more specific discussion of the unstack procedure.

The indirect method is almost always easier to use than the direct approach and has the advantage of not requiring that the stacked data be balanced. It has the disadvantage of using more computer memory since EViews must have two copies of the source data in memory at the same time.

Working with Pooled Data

The underlying series for each cross-section member are ordinary series, so all of the EViews tools for working with the individual cross-section series are available. In addition, EViews provides you with a number of specialized tools which allow you to work with your pool data. Using EViews, you can perform, in a single step, similar operations on all the series corresponding to a particular pooled variable.

Generating Pooled Data

You can generate or modify pool series using the pool series genr procedure. Click on **Pool-Genr** on the pool toolbar and enter a formula as you would for a regular genr, using pool series names as appropriate. Using our example from above, entering:

```
ratio? = g?/g_usa
```

is equivalent to entering the following four commands:

```
ratio_usa = g_usa/g_usa
ratio_uk = g_uk/g_usa
ratio_jpn = g_jpn/g_usa
ratio_kor = g_kor/g_usa
```

Generation of a pool series applies the formula you supply using an implicit loop across cross-section identifiers, creating or modifying one or more series as appropriate.

You may use pool and ordinary genr together to generate new pool variables. For example, to create a dummy variable that is equal to 1 for the US and 0 for all other countries, first select **PoolGenr** and enter:

```
dum? = 0
```

to initialize all four of the dummy variable series to 0. Then, to set the US values to 1, select **Quick/Generate Series...** from the main menu, and enter:

```
dum_usa = 1
```

It is worth pointing out that a superior method of creating this pool series is to use @GROUP to define a group called US containing only the "_USA" identifier (see "Group Definitions" on page 828), then to use the @INGRP function:

```
dum? = @ingrp(us)
```

to generate and implicitly refer to the four series (see "Pool Series" on page 830).

To modify a set of series using a pool, select **PoolGenr**, and enter the new pool series expression:

```
dum? = dum? * (g? > c?)
```

It is worth the reminder that the method used by the pool genr is to perform an implicit loop across the cross-section identifiers. This implicit loop may be exploited in various ways, for example, to perform calculations across cross-sectional units in a given period. Suppose, we have an *ordinary* series SUM which is initialized to zero. The pool genr expression:

```
sum = sum + c?
```

is equivalent to the following four ordinary genr statements:

```
sum = sum + c_usa
sum = sum + c_uk
sum = sum + c_jpn
sum = sum + c_kor
```

Bear in mind that this example is provided merely to illustrate the notion of implicit looping, since EViews provides built-in features to compute period-specific statistics.

Examining Your Data

Pool workfiles provide you with the flexibility to examine cross-section specific series as individual time series or as part of a larger set of series.

Examining Unstacked Data

Simply open an individual series and work with it using the standard tools available for examining a series object. Or create a group of series and work with the tools for a group object. One convenient way to create groups of series is to use tools for creating groups out of pool and ordinary series; another is to use wildcards expressions in forming the group.

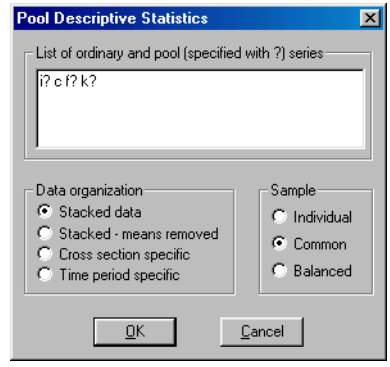
Examining Stacked Data

As demonstrated in "Stacked Data" beginning on page 833, you may use your pool object to view your data in stacked spreadsheet form. Select **View/Spreadsheet View...**, and list the series you wish to display. The names can include both ordinary and pool series names. Click on the **Order** + /– button to toggle between stacking your observations by cross-section and by date.

We emphasize that stacking your data only provides an alternative view of the data, and does not change the structure of the individual series in your workfile. Stacking data is not necessary for any of the data management or estimation procedures described below.

Calculating Descriptive Statistics

EViews provides convenient built-in features for computing various descriptive statistics for pool series using a pool object. To display the **Pool Descriptive Statistics** dialog, select **View/Descriptive Statistics...** from the pool toolbar.



In the edit box, you should list the ordinary and pooled series for which you want to compute the descriptive statistics. EViews will compute the mean, median, minimum, maximum, standard deviation, skewness, kurtosis, and the Jarque-Bera statistic for these series.

First, you should choose between the three sample options on the right of the dialog:

- **Individual:** uses the maximum number of observations available. If an observation on a variable is available for a particular cross-section, it is used in computation.
- **Common:** uses an observation only if data on the variable are available for all cross-sections in the same period. This method is equivalent to performing listwise exclusion by variable, then cross-sectional casewise exclusion within each variable.
- **Balanced**: includes observations when data on all variables in the list are available for all cross-sections in the same period. The balanced option performs casewise exclusion by both variable and cross-section.

Next, you should choose the computational method corresponding to one of the four data structures:

- **Stacked data**: display statistics for each variable in the list, computed over all cross-sections and periods. These are the descriptive statistics that you would get if you ignored the pooled nature of the data, stacked the data, and computed descriptive statistics.
- **Stacked – means removed**: compute statistics for each variable in the list after removing the cross-sectional means, taken over all cross-sections and periods.
- **Cross-section specific**: show the descriptive statistics for each cross-sectional variable, computed across all periods. These are the descriptive statistics derived by computing statistics for the individual series.
- **Time period specific**: compute period-specific statistics. For each period, compute the statistic using data on the variable from all the cross-sectional units in the pool.

Click on **OK**, and EViews will display a pool view containing tabular output with the requested statistics. If you select **Stacked data** or **Stacked - means removed**, the view will show a single column containing the descriptive statistics for each ordinary and pool series in the list, computed from the stacked data. If you select **Cross-section specific**, EViews will show a single column for each ordinary series, and multiple columns for each pool series. If you select **Time period specific**, the view will show a single column for each ordinary or pool series statistic, with each row of the column corresponding to a period in the workfile. Note that there will be a separate column for each statistic computed for an ordinary or pool series; a column for the mean, a column for the variance, *etc*.

You should be aware that the latter two methods may produce a great deal of output. Cross-section specific computation generates a set of statistics for each pool series/cross-section combination. If you ask for statistics for three pool series and there are 20 cross-sections in your pool, EViews will display 60 columns of descriptive statistics. For time period specific computation, EViews computes a set of statistics for each date/series combination. If you have a sample with 100 periods and you provide a list of three pool series, EViews will compute and display a view with columns corresponding to 3 sets of statistics, each of which contains values for 100 periods.

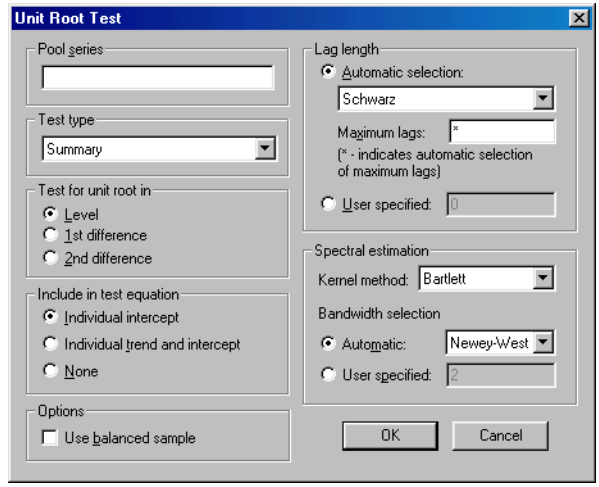
If you wish to compute period-specific statistics, you may alternatively elect to save the results in series objects. See "Making Period Stats" on page 842.

Computing Unit Root Tests

EViews provides convenient tools for computing multiple-series unit root tests for pooled data using a pool object. You may use the pool to compute one or more of the following types of unit root tests: Levin, Lin and Chu (2002), Breitung (2002), Im, Pesaran and Shin (2003), Fisher-type tests using ADF and PP tests—Maddala and Wu (1999) and Choi (2001), and Hadri (1999).

To compute the unit root test, select **View/Unit Root Test...**from the menu of a pool object.

Enter the name of an ordinary or pool series in the topmost edit field, then specify the remaining settings in the dialog.



These tests, along with the settings in the dialog, are described in considerable detail in "Panel Unit Root Tests" on page 530.

Making a Group of Pool Series

If you click on **Proc/Make Group...** and enter the names of ordinary and pool series. EViews will use the pool definitions to create an untitled group object containing the specified series. This procedure is useful when you wish to work with a set of pool series using the tools provided for groups.

Suppose, for example, that you wish to compute the covariance matrix for the C? series. Simply open the **Make Group** dialog, and enter the pool series name C?. EViews will create a group containing the set of cross-section specific series, with names beginning with "C" and ending with a cross-section identifier.

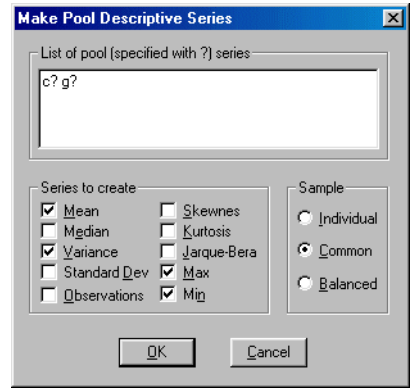
Then, in the new group object, you may select **View/Covariances** and either **Common Sample** or **Pairwise Sample** to describe the handling of missing values. EViews will display a view of the covariance matrix for all of the individual series in the group. Each element represents a covariance between individual members of the set of series in the pool series C?.

Making Period Stats

To save period-specific statistics in series in the workfile, select **Proc/Make Period Stat Series...** from the pool window, and fill out the dialog.

In the edit window, list the series for which you wish to calculate period-statistics. Next, select the particular statistics you wish to compute, and choose a sample option.

EViews will save your statistics in new series and will open an untitled group window to display the results. The series will be named automatically using the base name followed by the name of the statistic (MEAN, MED, VAR, SD, OBS, SKEW, KURT, JARQ, MAX, MIN). In this example, EViews will save the statistics using the names CMEAN, GMEAN, CVAR, GVAR, CMAX, GMAX, CMIN, and GMIN.

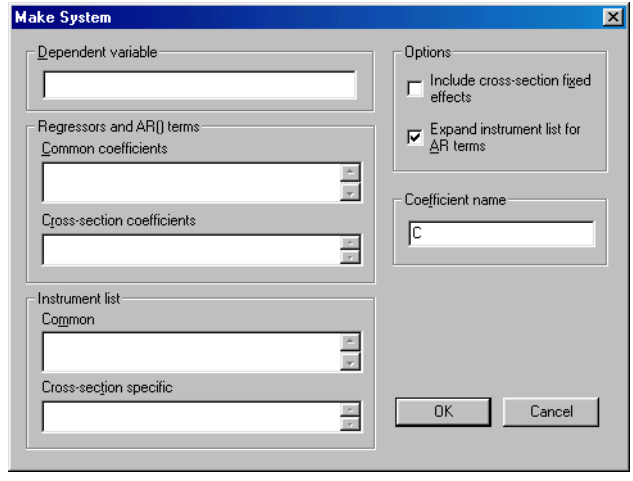


Making a System

Suppose that you wish to estimate a complex specification that cannot easily be estimated using the built-in features of the pool object. For example, you may wish to estimate a pooled equation imposing arbitrary coefficient restrictions, or using specialized GMM techniques that are not available in pooled estimation.

In these circumstances, you may use the pool to create a system object using both common and cross-section specific coefficients, AR terms, and instruments. The resulting system object may then be further customized, and estimated using all of the techniques available for system estimation.

Select **Proc/Make System…** and fill out the dialog. You may enter the dependent variable, common and cross-section specific variables, and use the checkbox to allow for cross-sectional fixed effects. You may also enter a list of common and cross-section specific instrumental variables, and instruct EViews to add lagged dependent and independent regressors as instruments in models with AR specifications.



When you click on **OK**, EViews will take your specification and create a new system object containing a single equation for each cross-section, using the specification provided.

Deleting/Storing/Fetching Pool Data

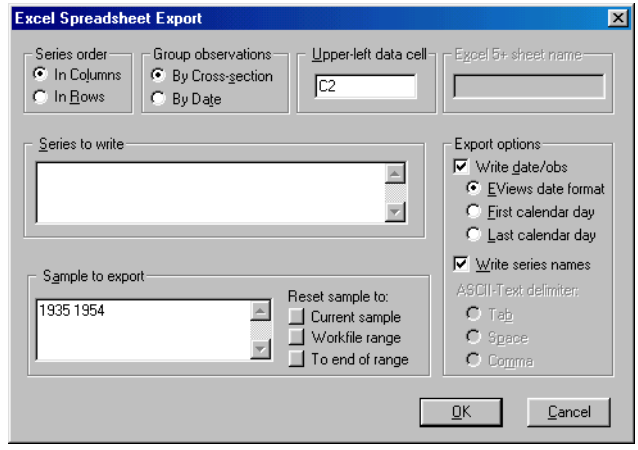
Pools may be used to delete, store, or fetch sets of series. Simply select **Proc/Delete pool series…**, **Proc/Store pool series (DB)…**, or **Proc/Fetch pool series (DB)…** as appropriate, and enter the ordinary and pool series names of interest.

If, for example, you instruct EViews to delete the pool series C?, EViews will loop through all of the cross-section identifiers and delete all series whose names begin with the letter "C" and end with the cross-section identifier.

Exporting Pooled Data

You can export your data into a disk file, or into a new workfile or workfile page, by reversing one of the procedures described above for data input.

To write pooled data in stacked form into an ASCII text, Excel, or Lotus worksheet file, first open the pool object, then from the pool menu, select **Proc/Import Pool data (ASCII, .XLS, .WK?)…**. Note that in order to access the pool specific export tools, you must select this procedure from the pool menu, not from the workfile menu.

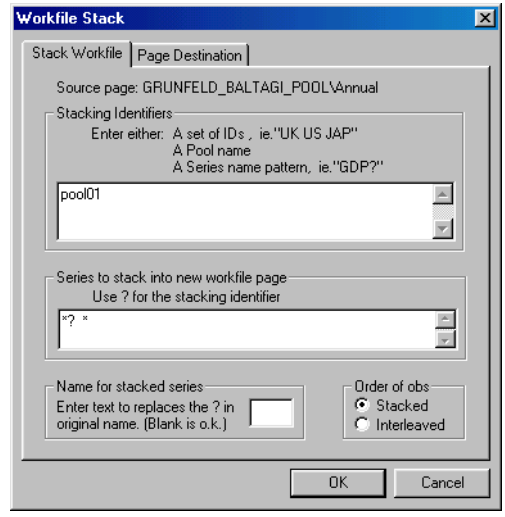


EViews will first open a file dialog prompting you to specify a file name and type. If you provide a new name, EViews will create the file; otherwise it will prompt you to overwrite the existing file.

Once you have specified your file, a pool write dialog will be displayed. Here we see the **Excel Spreadsheet Export** dialog. Specify the format of your data, including whether to write series in columns or in rows, and whether to stack by cross-section or by period. Then list the ordinary series, groups, and pool series to be written to the file, the sample of observations to be written, and select any export options. When you click on OK, EViews will write the specified file.

Since EViews allows you to both read and write data that are unstacked, stacked by cross-section, or stacked by date, you may use the pool import and export procedures to restructure your data in accordance with your needs.

Alternatively, you may use the workfile reshaping tools to stack the pooled data in a new workfile page. From the main workfile menu, select **Proc/Reshape Current Page/Stack in New Page...** to open the **Workfile Stack** dialog, and enter the name of a pool object in the top edit field, and the names of the ordinary series, groups, and pool series to be stacked in the second edit field.



The **Order of obs** option allows you to order the data in **Stacked** form (stacking the data by series, which orders by cross-section), or in **Interleaved** format (stacked the data by interleaving series, which orders the data by period or date).

The default naming rule for series in the destination is to use the base name. For example, if you stack the pool series "SALES?" and the individual series GENDER, the corresponding stacked series will, by default, be named "SALES", and "GENDER". If use of the default naming convention will create problems in the destination workfile, you should use the **Name for stacked series** field to specify an alternative. If, for example, you enter "_NEW", the target names will be formed by taking the base name, and appending the additional text, as in "SALES_NEW", and "GENDER_NEW".

See "Stacking a Workfile" on page 251 for a more detailed discussion of the workfile stacking procedure.

Pooled Estimation

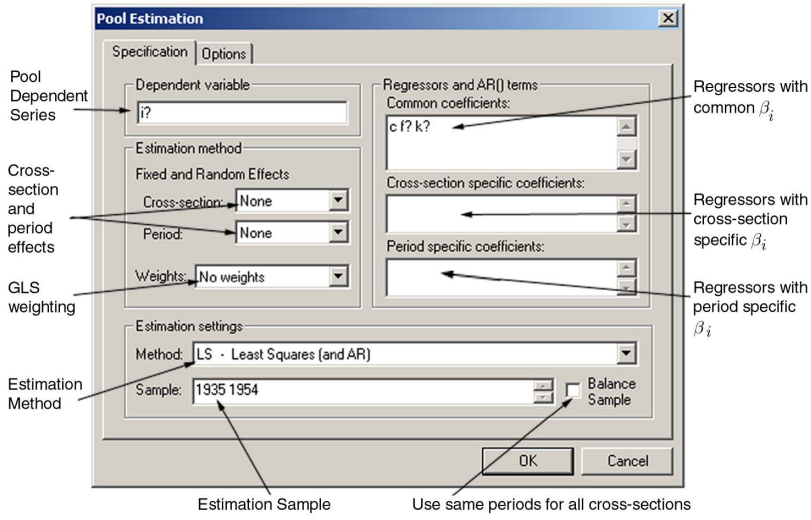
EViews pool objects allow you to estimate your model using least squares or instrumental variables (two-stage least squares), with correction for fixed or random effects in both the cross-section and period dimensions, AR errors, GLS weighting, and robust standard errors, all without rearranging or reordering your data.

We begin our discussion by walking you through the steps that you will take in estimating a pool equation. The wide range of models that EViews supports means that we cannot exhaustively describe all of the settings and specifications. A brief background discussion

of the supported techniques is provided in "Estimation Background" beginning on page 859.

Estimating a Pool Equation

To estimate a pool equation specification, simply press the **Estimate** button on your pool object toolbar or select **Proc/Estimate...** from the pool menu, and the basic pool estimation dialog will open:



First, you should specify the estimation settings in the lower portion of the dialog. Using the **Method** combo box, you may choose between **LS - Least Squares (and AR),** ordinary least squares regression, **TSLS - Two-Stage Least Squares (and AR)**, two-stage least squares (instrumental variable) regression. If you select the latter, the dialog will differ slightly from this example, with the provision of an additional tab (page) for you to specify your instruments (see "Instruments" on page 851).

You should also provide an estimation sample in the **Sample** edit box. By default, EViews will use the specified sample string to form use the largest sample possible in each cross-section. An observation will be excluded if any of the explanatory or dependent variables *for that cross-section* are unavailable in that period.

The checkbox for **Balanced Sample** instructs EViews to perform listwise exclusion over all cross-sections. EViews will eliminate an observation if data are unavailable *for any cross-section* in that period. This exclusion ensures that estimates for each cross-section will be based on a common set of dates.

Note that if all of the observations for a cross-section unit are not available, that unit will temporarily be removed from the pool for purposes of estimation. The EViews output will inform you if any cross-section were dropped from the estimation sample.

You may now proceed to fill out the remainder of the dialog.

Dependent Variable

List a pool variable, or an EViews expression containing ordinary and pool variables, in the **Dependent Variable** edit box.

Regressors and AR terms

On the right-hand side of the dialog, you should list your regressors in the appropriate edit boxes:

- **Common coefficients:** — enter variables that have the same coefficient across all cross-section members of the pool. EViews will include a single coefficient for each variable, and will label the output using the original expression.
- **Cross-section specific coefficients: —** list variables with different coefficients for each member of the pool. EViews will include a different coefficient for each cross-sectional unit, and will label the output using a combination of the cross-section identifier and the series name.
- **Period specific coefficients: —** list variables with different coefficients for each observed period. EViews will include a different coefficient for each period unit, and will label the output using a combination of the period identifier and the series name.

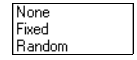
For example, if you include the ordinary variable TIME and POP? in the common coefficient list, the output will include estimates for TIME and POP?. If you include these variables in the cross-section specific list, the output will include coefficients labeled "_USA—TIME", "_UK—TIME", and "_USA—POP_USA", "_UK—POP_UK", etc.

Be aware that estimating your model with cross-section or period specific variables may generate large numbers of coefficients. If there are cross-section specific regressors, the number of these coefficients equals the product of the number of pool identifiers and the number of variables in the list; if there are period specific regressors, the number of corresponding coefficients is the number of periods times the number of variables in the list.

You may include AR terms in either the common or cross-section coefficients lists. If the terms are entered in the common coefficients list, EViews will estimate the model assuming a common AR error. If the AR terms are entered in the cross-section specific list, EViews will estimate separate AR terms for each pool member. See "Estimating AR Models" on page 497 for a description of AR specifications.

Note that EViews only allows specification by list for pool equations. If you wish to estimate a nonlinear specification, you must first create a system object, and then edit the system specification (see "Making a System" on page 843).

Fixed and Random Effects



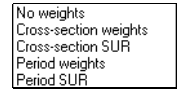
You should account for individual and period effects using the **Fixed and Random Effects** combo boxes. By default, EViews assumes that there are no effects so that the combo boxes are both set to **None**. You may change the default settings to allow for either **Fixed** or **Random** effects in either the cross-section or period dimension, or both.

There are some specifications that are not currently supported. You may not, for example, estimate random effects models with cross-section specific coefficients, AR terms, or weighting. Furthermore, while two-way random effects specifications are supported for balanced data, they may not be estimated in unbalanced designs.

Note that when you select a fixed or random effects specification, EViews will automatically add a constant to the common coefficients portion of the specification if necessary, to ensure that the effects sum to zero.

Weights

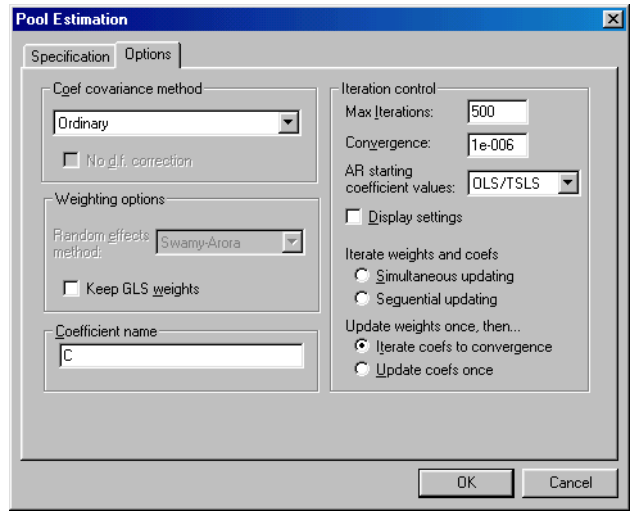
By default, all observations are given equal weight in estimation. You may instruct EViews to estimate your specification with estimated GLS weights using the combo box labeled **Weights**.



If you select **Cross section weights**, EViews will estimate a feasible GLS specification assuming the presence of cross-section heteroskedasticity. If you select **Cross-section SUR**, EViews estimates a feasible GLS specification correcting for both cross-section heteroskedasticity and contemporaneous correlation. Similarly, **Period weights** allows for period heteroskedasticity, while **Period SUR** corrects for both period heteroskedasticity and general correlation of observations within a given cross-section. Note that the SUR specifications are each examples of what is sometimes referred to as the Parks estimator.

Options

Clicking on the **Options** tab in the dialog brings up a page displaying a variety of estimation options for pool estimation. Settings that are not currently applicable will be grayed out.

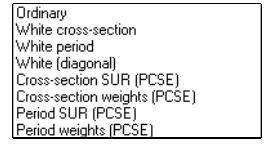


Coef Covariance Method

By default, EViews reports conventional estimates of coefficient standard errors and covariances.

You may use the combo box at the top of the page to select from the various robust methods available for computing the coefficient standard errors. Each of the methods is described in greater detail in "Robust Coefficient Covariances" on page 869.

Note that the checkbox **No d.f. correction** permits to you compute robust covariances without the leading degree of freedom correction term. This option may make it easier to match EViews results to those from other sources.



Weighting Options

If you are estimating a specification that includes a random effects specification, EViews will provide you with a **Random effects method** combo box so that you may specify one of the methods for calculating estimates of the component variances. You may choose between the default **Swamy-Arora, Wallace-Hussain**, or **Wansbeek-Kapteyn** methods. See "Random Effects" on page 863 for discussion of the differences between the methods. Note that the default Swamy-Arora method should be the most familiar from textbook discussions.

Details on these methods are provided in Baltagi (2001), Baltagi and Chang (1994), Wansbeek and Kapteyn (1989).

The checkbox labeled **Keep GLS weights** may be selected to require EViews to save all estimated GLS weights with the equation, regardless of their size. By default, EViews will not save estimated weights in system (SUR) settings, since the size of the required matrix may be quite large. If the weights are not saved with the equation, there may be some pool views and procedures that are not available.

Coefficient Name

By default, EViews uses the default coefficient vector C to hold the estimates of the coefficients and effects. If you wish to change the default, simply enter a name in the edit field. If the specified coefficient object exists, it will be used, after resizing if necessary. If the object does not exist, it will be created with the appropriate size. If the object exists but is an incompatible type, EViews will generate an error.

Iteration Control

The familiar **Max Iterations** and **Convergence** criterion edit boxes that allow you to set the convergence test for the coefficients and GLS weights.

If your specification contains AR terms, the **AR starting coefficient values** combo box allows you to specify starting values as a fraction of the OLS (with no AR) coefficients, zero, or user-specified values.

If **Display Settings** is checked, EViews will display additional information about convergence settings and initial coefficient values (where relevant) at the top of the regression output.

The last set of radio buttons is used to determine the iteration settings for coefficients and GLS weighting matrices.

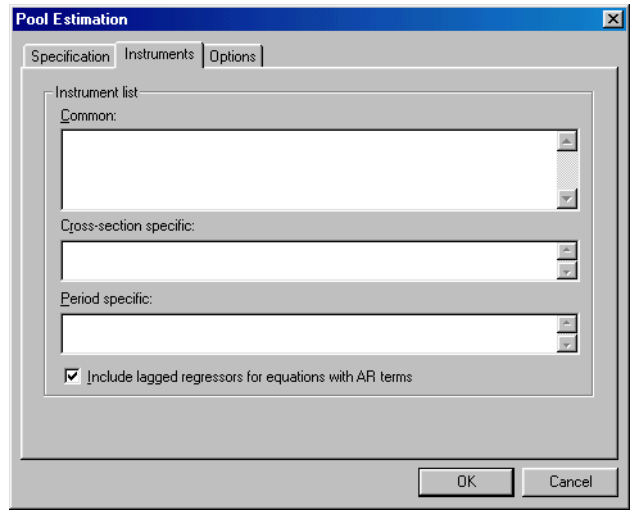
The first two settings, **Simultaneous updating** and **Sequential updating** should be employed when you want to ensure that both coefficients and weighting matrices are iterated to convergence. If you select the first option, EViews will, at every iteration, update both the coefficient vector and the GLS weights; with the second option, the coefficient vector will be iterated to convergence, then the weights will be updated, then the coefficient vector will be iterated, and so forth. Note that the two settings are identical for GLS models without AR terms.

If you select one of the remaining two cases, **Update coefs to convergence** and **Update coefs once**, the GLS weights will only be updated once. In both settings, the coefficients are first iterated to convergence, if necessary, in a model with no weights, and then the weights are computed using these first-stage coefficient estimates. If the first option is selected, EViews will then iterate the coefficients to convergence in a model that uses the first-stage weight estimates. If the second option is selected, the first-stage coefficients will only be iterated once. Note again that the two settings are identical for GLS models without AR terms.

By default, EViews will update GLS weights once, and then will update the coefficients to convergence.

Instruments

To estimate a pool specification using instrumental variables techniques, you should select **TSLS - Two-Stage Least Squares (and AR)** in the **Method** combo box at the bottom of the main (**Specification**) dialog page. EViews will respond by creating a three-tab dialog in which the middle tab (page) is used to specify your instruments.



As with the regression specification, the instrument list specification is divided into a set of **Common**, **Cross-section specific**, and **Period specific** instruments. The interpretation of these lists is the same as for the regressors; if there are cross-section specific instruments, the number of these instruments equals the product of the number of pool identifiers and the number of variables in the list; if there are period specific instruments, the number of corresponding instruments is the number of periods times the number of variables in the list.

Note that you need not specify constant terms explicitly since EViews will internally add constants to the lists corresponding to the specification in the main page.

Lastly, there is a checkbox labeled **Include lagged regressors for equations with AR terms** that will be displayed if your specification includes AR terms. Recall that when estimating an AR specification, EViews performs nonlinear least squares on an AR differenced specification. By default, EViews will add lagged values of the dependent and independent regressors to the corresponding lists of instrumental variables to account for the modified differenced specification. If, however, you desire greater control over the set of instruments, you may uncheck this setting.

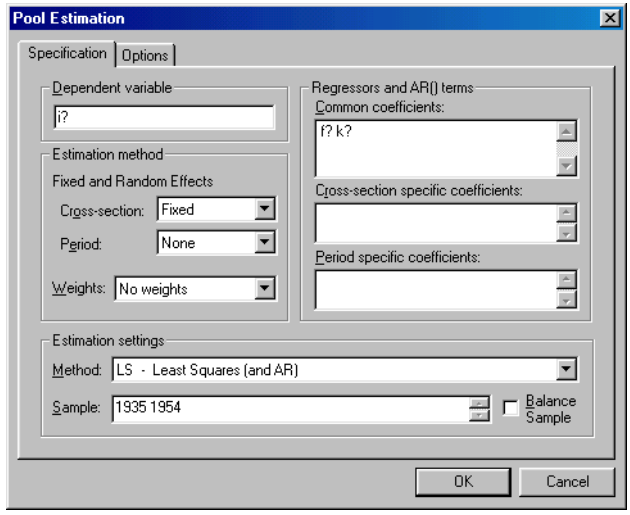
Pool Equation Examples

For illustrative purposes, we employ the balanced firm-level data from Grunfeld (1958) that have been used extensively as an example dataset (*e.g.*, Baltagi, 2001). The panel data consist of annual observations on investement (I?), firm value (F?), and capital stock (K?) for 10 large U.S. manufacturing firms for the 20 years from 1935-54.

The pool identifiers for our data are “AR”, “CH”, “DM”, “GE”, “GM”, “GY”, “IB”, “UO”, “US”, “WH”.

We cannot possibly demonstrate all of the specifications that may be estimated using these data, but we provide a few illustrative examples.

First, we estimate a model regressing I? on the common regressors F? and K?, with a cross-section fixed effect. All regression coefficients are restricted to be the same across all cross-sections, so this is equivalent to estimating a model on the stacked data, using the cross-sectional identifiers only for the fixed effect.



The top portion of the output from this regression, which shows the dependent variable, method, estimation and sample information is given by:

Dependent Variable: I?
Method: Pooled Least Squares
Date: 12/03/03 Time: 12:21
Sample: 1935 1954
Included observations: 20
Number of cross-sections used: 10
Total pool (balanced) observations: 200

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-58.74394	12.45369	-4.716990	0.0000
F?	0.110124	0.011857	9.287901	0.0000
K?	0.310065	0.017355	17.86656	0.0000
Fixed Effects (Cross)				
AR--C	-55.87287			
CH--C	30.93464			
DM--C	52.17610			
GE--C	-176.8279			
GM--C	-11.55278			
GY--C	-28.47833			
IB--C	35.58264			
UO--C	-7.809534			
US--C	160.6498			
WH--C	1.198282			

EViews displays both the estimates of the coefficients and the fixed effects. Note that EViews automatically includes a constant term so that the fixed effects estimates sum to zero and should be interpreted as deviations from an overall mean.

Note also that the estimates of the fixed effects do not have reported standard errors since EViews treats them as nuisance parameters for the purposes of estimation. If you wish to compute standard errors for the cross-section effects, you may estimate a model without a constant and explicitly enter the C in the **Cross-section specific coefficients** edit field.

The bottom portion of the output displays the effects specification and summary statistics for the estimated model.

Effects Specification			
Cross-section fixed (dummy variables)			
R-squared	0.944073	Mean dependent var	145.9583
Adjusted R-squared	0.940800	S.D. dependent var	216.8753
S.E. of regression	52.76797	Akaike info criterion	10.82781
Sum squared resid	523478.1	Schwarz criterion	11.02571
Log likelihood	-1070.781	F-statistic	288.4996
Durbin-Watson stat	0.716733	Prob(F-statistic)	0.000000

A few of these summary statistics require discussion. First, the reported R-squared and F-statistics are based on the difference between the residuals sums of squares from the estimated model, and the sums of squares from a *single* constant-only specification, not from a fixed-effect-only specification. As a result, the interpretation of these statistics is that they describe the explanatory power of the entire specification, including the estimated fixed effects. Second, the reported information criteria use, as the number of parameters, the number of estimated coefficients, including fixed effects. Lastly, the reported Durbin-Watson stat is formed simply by computing the first-order residual correlation on the stacked set of residuals.

We may reestimate this specification using White cross-section standard errors to allow for general contemporaneous correlation between the firm residuals. The "cross-section" designation is used to indicate that covariances are allowed across cross-sections (contemporaneously). Simply click on the **Options** tab and select **White cross-section** as the coefficient covariance matrix, then reestimate the model. The relevant portion of the output is given by:

White cross-section standard errors & covariance (d.f. corrected)

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-58.74394	19.61460	-2.994909	0.0031
F?	0.110124	0.016932	6.504061	0.0000
K?	0.310065	0.031541	9.830701	0.0000

The new output shows the method used for computing the standard errors, and the new standard error estimates, t-statistic values, and probabilities reflecting the robust calculation of the coefficient covariances.

Alternatively, we may adopt the Arellano (1987) approach of computing White coefficient covariance estimates that are robust to arbitrary within cross-section residual correlation. Select the **Options** page and choose **White period** as the coefficient covariance method.

We caution, however, that these results assume that the number of cross-sections is large, which is not the case.

White period standard errors & covariance (d.f. corrected)

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-58.74394	26.87312	-2.185974	0.0301
F?	0.110124	0.014793	7.444423	0.0000
K?	0.310065	0.051357	6.037432	0.0000

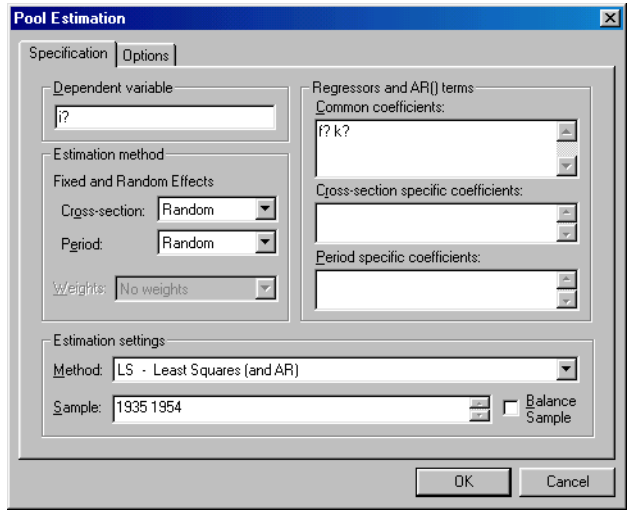
We may add an AR(1) term to the specification, and compute estimates using Period SUR PCSE methods to compute standard errors that are robust to more general serial correlation. EViews will estimate the transformed model using nonlinear least squares, will form an estimate of the residual covariance matrix, and will use the estimate in forming standard errors. The top portion of the results is given by:

Dependent Variable: I?
Method: Pooled Least Squares
Date: 12/03/03 Time: 14:14
Sample (adjusted): 1936 1954
Included observations: 19 after adjusting endpoints
Number of cross-sections used: 10
Total pool (balanced) observations: 190
Period SUR (PCSE) standard errors & covariance (d.f. corrected)
Convergence achieved after 12 iterations

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-63.45169	18.10027	-3.505566	0.0006
F?	0.094744	0.010226	9.264956	0.0000
K?	0.350205	0.044842	7.809792	0.0000
AR(1)	0.686108	0.073125	9.382711	0.0000

Note in particular the description of the sample adjustment where we show that the estimation drops one observation for each cross-section when performing the AR differencing, as well as the description of the method used to compute coefficient covariances.

Alternatively, we may produce estimates for the two way random effects specification. First, in the **Specification** page, we set both the cross-section and period effects combo boxes to **Random**. Note that the dialog changes to show that weighted estimation is not available with random effects (nor is AR estimation).



Next, in the **Options** page we change the **Random effects method** to use the **Wansbeek-Kapteyn** method of computing the estimates of the random component variances.

Lastly, we click on **OK** to estimate the model.

The top portion of the dialog displays basic information about the specification, including the method used to compute the component variances, as well as the coefficient estimates and associated statistics:

Dependent Variable: I?
Method: Pooled EGLS (Two-way random effects)
Date: 12/03/03 Time: 14:28
Sample: 1935 1954
Included observations: 20
Number of cross-sections used: 10
Total pool (balanced) observations: 200
Wansbeek and Kapteyn estimator of component variances

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-63.89217	30.53284	-2.092573	0.0377
F?	0.111447	0.010963	10.16577	0.0000
K?	0.323533	0.018767	17.23947	0.0000

The middle portion of the output (not depicted) displays the best-linear unbiased predictor estimates of the random effects themselves.

The next portion of the output describes the estimates of the component variances:

Effects Specification		
Cross-section random S.D. / Rho	89.26257	0.7315
Period random S.D. / Rho	15.77783	0.0229
Idiosyncratic random S.D. / Rho	51.72452	0.2456

Here, we see that the estimated cross-section, period, and idiosyncratic error component standard deviations are 89.26, 15.78, and 51.72, respectively. As seen from the values of Rho, these components comprise 0.73, 0.02 and 0.25 of the total variance. Taking the cross-section component, for example, Rho is computed as:

$$0.7315 \ = \ 89.26257^2/(89.26257^2 + 15.77783^2 + 51.72452^2) \tag{27.1}$$

In addition, EViews reports summary statistics for the random effects GLS weighted data used in estimation, and a subset of statistics computed for the unweighted data.

Suppose instead that we elect to estimate a specification with I? as the dependent variable, C and F? as the common regressors, and K? as the cross-section specific regressor, using cross-section weighted least squares. The top portion of the output is given by:

Dependent Variable: I?
Method: Pooled EGLS (Cross-section weights)
Date: 12/18/03 Time: 14:40
Sample: 1935 1954
Included observations: 20
Number of cross-sections used: 10
Total pool (balanced) observations: 200
Linear estimation after one-step weighting matrix

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-4.696363	1.103187	-4.257089	0.0000
F?	0.074084	0.004077	18.17140	0.0000
AR--KAR	0.092557	0.007019	13.18710	0.0000
CH--KCH	0.321921	0.020352	15.81789	0.0000
DM--KDM	0.434331	0.151100	2.874468	0.0045
GE--KGE	-0.028400	0.034018	-0.834854	0.4049
GM--KGM	0.426017	0.026380	16.14902	0.0000
GY--KGY	0.074208	0.007050	10.52623	0.0000
IB--KIB	0.273784	0.019948	13.72498	0.0000
UO--KUO	0.129877	0.006307	20.59268	0.0000
US--KUS	0.807432	0.074870	10.78444	0.0000
WH--KWH	-0.004321	0.031420	-0.137511	0.8908

Note that EViews displays results for each of the cross-section specific K? series, labeled using the equation identifier followed by the series name. For example, the coefficient labeled "AR--KAR" is the coefficient of KAR in the cross-section equation for firm AR.

In our last example, we consider the use of the @INGRP pool function to estimate an specification containing group dummy variables (see "Pool Series" on page 830). Suppose we modify our pool definition so that we have define a group named "MYGROUP" containing the identifiers "GE", "GM", and "GY". We may then estimate a pool specification using the common regressor list:

```
c f? k? @ingrp(mygrp)
```

where the latter pool series expression refers to a set of 10 implicit series containing dummy variables for group membership. The implicit series associated with the identifiers "GE", "GM", and "GY" will contain the value 1, and the remaining seven series will contain the value 0.

The results from this estimation are given by:

Dependent Variable: I?
Method: Pooled Least Squares
Date: 12/18/03 Time: 13:29
Sample: 1935 1954
Included observations: 20
Number of cross-sections used: 10
Total pool (balanced) observations: 200

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-34.97580	8.002410	-4.370659	0.0000
F?	0.139257	0.005515	25.25029	0.0000
K?	0.259056	0.021536	12.02908	0.0000
@INGRP(MYGRP)	-137.3389	14.86175	-9.241093	0.0000

R-squared	0.869338	Mean dependent var	145.9583
Adjusted R-squared	0.867338	S.D. dependent var	216.8753
S.E. of regression	78.99205	Akaike info criterion	11.59637
Sum squared resid	1222990.	Schwarz criterion	11.66234
Log likelihood	-1155.637	F-statistic	434.6841
Durbin-Watson stat	0.356290	Prob(F-statistic)	0.000000

We see that the mean value of I? for the three groups is substantially lower than for the remaining groups, and that the difference is statistically significant at conventional levels.

Pool Equation Views and Procedures

Once you have estimated your pool equation, you may examine your output in the usual ways:

Representation

Select **View/Representations** to examine your specification. EViews estimates your pool as a system of equations, one for each cross-section unit.

Estimation Output

View/Estimation Output will change the display to show the results from the pooled estimation.

As with other estimation objects, you can examine the estimates of the coefficient covariance matrix by selecting **View/Coef Covariance Matrix**.

Testing

EViews allows you to perform coefficient tests on the estimated parameters of your pool equation. Select **View/Wald Coefficient Tests…** and enter the restriction to be tested. Additional tests are described in the panel discussion "Panel Equation Testing" on page 922

Residuals

You can view your residuals in spreadsheet or graphical format by selecting **View/Residuals/Table** or **View/Residuals/Graph**. EViews will display the residuals for each cross-sectional equation. Each residual will be named using the base name RES, followed by the cross-section identifier.

If you wish to save the residuals in series for later use, select **Proc/Make Resids**. This procedure is particularly useful if you wish to form specification or hypothesis tests using the residuals.

Residual Covariance/Correlation

You can examine the estimated residual contemporaneous covariance and correlation matrices. Select **View/Residual** and then either **Correlation Matrix** or **Covariance Matrix** to examine the appropriate matrix.

Forecasting

To perform forecasts using a pool equation you will first make a model. Select **Proc/Make Model** to create an untitled model object that incorporates all of the estimated coefficients. If desired, this model can be edited. Solving the model will generate forecasts for the dependent variable for each of the cross-section units. For further details, see Chapter 26, "Models", on page 777.

Estimation Background

The basic class of models that can be estimated using a pool object may be written as:

$$Y_{it} = \alpha + X_{it}'\beta_{it} + \delta_i + \gamma_t + \epsilon_{it}, \tag{27.2}$$

where Y_{it} is the dependent variable, and X_{it} is a k-vector of regressors, and ϵ_{it} are the error terms for $i = 1, 2, \ldots, M$ cross-sectional units observed for dated periods $t = 1, 2, \ldots, T$. The α parameter represents the overall constant in the model, while the δ_i and γ_t represent cross-section or period specific effects (random or fixed). Identification obviously requires that the β coefficients have restrictions placed upon them. They may be divided into sets of common (across cross-section and periods), cross-section specific, and period specific regressor parameters.

While most of our discussion will be in terms of a balanced sample, EViews does not require that your data be balanced; missing values may be used to represent observations that are not available for analysis in a given period. We will detail the unbalanced case only where deemed necessary.

We may view these data as a set of cross-section specific regressions so that we have M cross-sectional equations each with T observations stacked on top of one another:

$$Y_i = \alpha l_T + X_i'\beta_{it} + \delta_i l_T + I_T\gamma + \epsilon_i \tag{27.3}$$

for $i = 1, \ldots, M$, where l_T is a T-element unit vector, I_T is the T-element identity matrix, and γ is a vector containing all of the period effects, $\gamma' = (\gamma_1, \gamma_2, \ldots, \gamma_T)$.

Analogously, we may write the specification as a set of T period specific equations, each with M observations stacked on top of one another.

$$Y_t = \alpha l_M + X_t'\beta_{it} + I_M\delta + \gamma_t l_M + \epsilon_t \tag{27.4}$$

for $t = 1, \ldots, T$, where l_M is a M-element unit vector, I_M is the M-element identity matrix, and δ is a vector containing all of the cross-section effects, $\delta' = (\delta_1, \delta_2, \ldots, \delta_M)$.

For purposes of discussion we will employ the stacked representation of these equations. First, for the specification organized as a set of cross-section equations, we have:

$$Y = \alpha l_{MT} + X\beta + (I_M \otimes l_T)\delta + (l_M \otimes I_T)\gamma + \epsilon \tag{27.5}$$

where the matrices β and X are set up to impose any restrictions on the data and parameters between cross-sectional units and periods, and where the general form of the unconditional error covariance matrix is given by:

$$\Omega = E(\epsilon\epsilon') = E\begin{pmatrix} \epsilon_1\epsilon_1' & \epsilon_2\epsilon_1' & \cdots & \epsilon_M\epsilon_1' \\ \epsilon_2\epsilon_1' & \epsilon_2\epsilon_2' & & \vdots \\ & & \ddots & \\ \epsilon_M\epsilon_1' & \cdots & & \epsilon_M\epsilon_M' \end{pmatrix} \tag{27.6}$$

If instead we treat the specification as a set of period specific equations, the stacked (by period) representation is given by,

$$Y = \alpha l_{MT} + X\beta + (l_M \otimes I_T)\delta + (I_M \otimes l_T)\gamma + \epsilon \tag{27.7}$$

with error covariance,

$$\Omega = E(\epsilon\epsilon') = E\begin{pmatrix} \epsilon_1\epsilon_1' & \epsilon_2\epsilon_1' & \cdots & \epsilon_T\epsilon_1' \\ \epsilon_2\epsilon_1' & \epsilon_2\epsilon_2' & & \vdots \\ & & \ddots & \\ \epsilon_T\epsilon_1' & \cdots & & \epsilon_T\epsilon_T' \end{pmatrix} \tag{27.8}$$

The remainder of this section describes briefly the various components that you may employ in an EViews pool specification.

Cross-section and Period Specific Regressors

The basic EViews pool specification in Equation (27.2) allows for β slope coefficients that are common to all individuals and periods, as well as coefficients that are either cross-section or period specific. Before turning to the general specification, we consider three extreme cases.

First, if all of the β_{it} are common across cross-sections and periods, we may simplify the expression for Equation (27.2) to:

$$Y_{it} = \alpha + X_{it}'\beta + \delta_i + \gamma_t + \epsilon_{it} \tag{27.9}$$

There are a total of k coefficients in β, each corresponding to an element of x.

Alternately, if all of the β_{it} coefficients are cross-section specific, we have:

$$Y_{it} = \alpha + X_{it}'\beta_i + \delta_i + \gamma_t + \epsilon_{it} \tag{27.10}$$

Note that there are k in each β_i for a total of Mk slope coefficients.

Lastly, if all of the β_{it} coefficients are period specific, the specification may be written as:

$$Y_{it} = \alpha + X_{it}'\beta_t + \delta_i + \gamma_t + \epsilon_{it} \tag{27.11}$$

for a total of Tk slope coefficients.

More generally, splitting X_{it} into the three groups (common regressors X_{0it}, cross-section specific regressors X_{1it}^2, and period specific regressors X_{2it}), we have:

$$Y_{it} = \alpha + X_{0it}'\beta_0 + X_{1it}'\beta_{1i} + X_{2it}'\beta_{2t} + \delta_i + \gamma_t + \epsilon_{it} \tag{27.12}$$

If there are k_1 common regressors, k_2 cross-section specific regressors, and k_3 period specific regressors, there are a total of $k_0 = k_1 + k_2M + k_3T$ regressors in β.

EViews estimates these models by internally creating interaction variables, M for each regressor in the cross-section regressor list and T for each regressor in the period-specific list, and using them in the regression. Note that estimating models with cross-section or period specific coefficients may lead to the generation of a large number of implicit interaction variables, and may be computationally intensive, or lead to singularities in estimation.

AR Specifications

EViews provides convenient tools for estimating pool specifications that include AR terms. Consider a restricted version of Equation (27.2) on page 860 that does not admit period specific regressors or effects,

$$Y_{it} = \alpha + X_{it}'\beta_i + \delta_i + \gamma_t + \epsilon_{it} \tag{27.13}$$

where the cross-section effect δ_i is either not present, or is specified as a fixed effect. We then allow the residuals to follow a general AR process:

$$\epsilon_{it} = \sum_{r=1}^{p} \rho_{ri}\epsilon_{it-r} + \eta_{it} \tag{27.14}$$

for all i, where the innovations η_{it} are independent and identically distributed, assuming further that there is no unit root. Note that we allow the autocorrelation coefficients ρ to be cross-section, but not period specific.

If, for example, we assume that ϵ_{it} follows an AR(1) process with cross-section specific AR coefficients, EViews will estimate the transformed equation:

$$Y_{it} = \rho_{1i}Y_{it-1} + \alpha(1-\rho_{1i}) + (X_{it} - \rho_{1i}X_{it-1})'\beta_i + \delta_i(1-\rho_{1i}) + \eta_{it} \tag{27.15}$$

using iterative techniques to estimate $(\alpha, \beta_i, \rho_i)$ for all i. See "Estimating AR Models" on page 497 for additional discussion.

We emphasize that EViews does place are restrictions on the specifications that admit AR errors. AR terms may not be estimated in specifications with period specific regressors or effects. Lastly, AR terms are not allowed in selected GLS specifications (random effects, period specific heteroskedasticity and period SUR). In those GLS specifications where AR terms are allowed, the error covariance assumption is for the innovations not the autoregressive error.

Fixed and Random Effects

The presence of cross-section and period specific effects terms δ and γ may be handled using fixed or random effects methods.

You may, with some restrictions, specify models containing effects in one or both dimension, for example, a fixed effect in the cross-section dimension, a random effect in the period dimension, or a fixed effect in the cross-section and a random effect in the period

dimension. Note, in particular, however, that two-way random effects may only be estimated if the data are balanced so that every cross-section has the same set of observations.

Fixed Effects

The fixed effects portions of specifications are handled using orthogonal projections. In the simple one-way fixed effect specifications and the balanced two-way fixed specification, these projections involve the familiar approach of removing cross-section or period specific means from the dependent variable and exogenous regressors, and then performing the specified regression on the demean (see, for example Baltagi, 2001). More generally, we apply the results from Davis (2002) for estimating multi-way error components models with unbalanced data.

Note that if instrumental variables estimation is specified with fixed effects, EViews will automatically add to the instrument list, the constants implied by the fixed effects so that the orthogonal projection is also applied to the instrument list.

Random Effects

The random effects specifications assumes that the corresponding effects δ_i and γ_t are realizations of independent random variables with mean zero and finite variance. Most importantly, the random effects specification assumes that the effect is uncorrelated with the idiosyncratic residual ϵ_{it}.

EViews handles the random effects models using feasible GLS techniques. The first step, estimation of the covariance matrix for the composite error formed by the effects and the residual (*e.g.*, $\nu_{it} = \delta_i + \gamma_t + \epsilon_{it}$ in the two-way random effects specification), uses one of the quadratic unbiased estimators (QUE) from Swamy-Arora, Wallace-Hussain, or Wansbeek-Kapteyn. Briefly, the three QUE methods use the expected values from quadratic forms in one or more sets of first-stage estimated residuals to compute moment estimates of the component variances $(\sigma_\delta^2, \sigma_\gamma^2, \sigma_\epsilon^2)$. The methods differ only in the specifications estimated in evaluating the residuals, and the resulting forms of the moment equations and estimators.

The Swamy-Arora estimator of the component variances, cited most often in textbooks, uses residuals from the within (fixed effect) and between (means) regressions. In contrast, the Wansbeek and Kapteyn estimator uses only residuals from the fixed effect (within) estimator, while the Wallace-Hussain estimator uses only OLS residuals. In general, the three should provide similar answers, especially in large samples. The Swamy-Arora estimator requires the calculation of an additional model, but has slightly simpler expressions for the component variance estimates. The remaining two may prove easier to estimate in some settings.

Additional details on random effects models are provided in Baltagi (2001), Baltagi and Chang (1994), Wansbeek and Kapteyn (1989). Note that your component estimates may

differ slightly from those obtained from other sources since EViews always uses the more complicated *unbiased* estimators involving traces of matrices that depend on the data (see Baltagi (2001) for discussion, especially "Note 2" on p. 27).

Once the component variances have been estimated, we form an estimator of the composite residual covariance, and then GLS transform the dependent and regressor data.

If instrumental variables estimation is specified with random effects, EViews will GLS transform both the data and the instruments prior to estimation. This approach to random effects estimation has been termed generalized two-stage least squares (G2SLS). See Baltagi (2001, pp. 111-115) and "Random Effects and GLS" on page 868 for additional discussion.

Generalized Least Squares

You may estimate GLS specifications that account for various patterns of correlation between the residuals. There are four basic variance structures that you may specify: cross-section specific heteroskedasticity, period specific heteroskedasticity, contemporaneous covariances, and between period covariances.

Note that all of the GLS specifications described below may be estimated in one-step form, where we estimate coefficients, compute a GLS weighting transformation, and then reestimate on the weighted data, or in iterative form, where to repeat this process until the coefficients and weights converge.

Cross-section Heteroskedasticity

Cross-section heteroskedasticity allows for a different residual variance for each cross section. Residuals between different cross-sections and different periods are assumed to be 0. Thus, we assume that:

$$\begin{aligned} E(\epsilon_{it}\epsilon_{it}|X_i^*) &= \sigma_i^2 \\ E(\epsilon_{is}\epsilon_{jt}|X_i^*) &= 0 \end{aligned} \tag{27.16}$$

for all i, j, s and t with $i \neq j$ and $s \neq t$, where X_i^* contains X_i and, if estimated by fixed effects, the relevant cross-section or period effects (δ_i, γ).

Using the cross-section specific residual vectors, we may rewrite the main assumption as:

$$E(\epsilon_i\epsilon_i'|X_i^*) = \sigma_i^2 I_T \tag{27.17}$$

GLS for this specification is straightforward. First, we perform preliminary estimation to obtain cross-section specific residual vectors, then we use these residuals to form estimates of the cross-specific variances. The estimates of the variances are then used in a weighted least squares procedure to form the feasible GLS estimates.

Period Heteroskedasticity

Exactly analogous to the cross-section case, period specific heteroskedasticity allows for a different residual variance for each period. Residuals between different cross-sections and different periods are still assumed to be 0 so that:

$$\begin{aligned} E(\epsilon_{it}\epsilon_{jt}|X_t^*) &= \sigma_t^2 \\ E(\epsilon_{is}\epsilon_{jt}|X_t^*) &= 0 \end{aligned} \tag{27.18}$$

for all i, j, s and t with $s \neq t$, where X_t^* contains X_t and, if estimated by fixed effects, the relevant cross-section or period effects (δ, γ_t).

Using the period specific residual vectors, we may rewrite the first assumption as:

$$E(\epsilon_t\epsilon_t'|X_t^*) = \sigma_t^2 I_M \tag{27.19}$$

We perform preliminary estimation to obtain period specific residual vectors, then we use these residuals to form estimates of the period variances, reweight the data, and then form the feasible GLS estimates.

Contemporaneous Covariances (Cross-section SUR)

This class of covariance structures allows for conditional correlation between the contemporaneous residuals for cross-section i and j, but restricts residuals in different periods to be uncorrelated. More specifically, we assume that:

$$\begin{aligned} E(\epsilon_{it}\epsilon_{jt}|X_t^*) &= \sigma_{ij} \\ E(\epsilon_{is}\epsilon_{jt}|X_t^*) &= 0 \end{aligned} \tag{27.20}$$

for all i, j, s and t with $s \neq t$. Note that the contemporaneous covariances do not vary over t.

Using the period specific residual vectors, we may rewrite this assumption as,

$$E(\epsilon_t\epsilon_t'|X_t^*) = \Omega_M \tag{27.21}$$

for all t, where,

$$\Omega_M = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1M} \\ \sigma_{12} & \sigma_{22} & & \vdots \\ & & \ddots & \\ \sigma_{M1} & \cdots & & \sigma_{MM} \end{pmatrix} \tag{27.22}$$

There does not appear to be a commonly accepted name for this variance structure, so we term it a *Cross-section SUR* specification since it involves covariances across cross-sections

as in a seemingly unrelated regressions type framework (where each equation corresponds to a cross-section).

Cross-section SUR weighted least squares on this specification (sometimes referred to as the Parks estimator) is simply the feasible GLS estimator for systems where the residuals are both cross-sectionally heteroskedastic and contemporaneously correlated. We employ residuals from first stage estimates to form an estimate of Ω_M. In the second stage, we perform feasible GLS.

Bear in mind that there are potential pitfalls associated with the SUR/Parks estimation (see Beck and Katz (1995)). For one, EViews may be unable to compute estimates for this model when you the dimension of the relevant covariance matrix is large and there are a small number of observations available from which to obtain covariance estimates. For example, if we have a cross-section SUR specification with large numbers of cross-sections and a small number of time periods, it is quite likely that the estimated residual correlation matrix will be nonsingular so that feasible GLS is not possible.

It is worth noting that an attractive alternative to the SUR methodology estimates the model without a GLS correction, then corrects the coefficient estimate covariances to account for the contemporaneous correlation. See "Robust Coefficient Covariances" on page 869.

Note also that if cross-section SUR is combined with instrumental variables estimation, EViews will employ a Generalized Instrumental Variables estimator in which both the data and the instruments are transformed using the estimated covariances. See Wooldridge (2002) for discussion and comparison with the three-stage least squares approach.

(Period Heteroskedasticity and Serial Correlation) Period SUR

This class of covariance structures allows for arbitrary period serial correlation and period heteroskedasticity between the residuals for a given cross-section, but restricts residuals in different cross-sections to be uncorrelated. Accordingly, we assume that:

$$\begin{aligned} E(\epsilon_{is}\epsilon_{it}|X_i^*) &= \sigma_{st} \\ E(\epsilon_{is}\epsilon_{jt}|X_i^*) &= 0 \end{aligned} \tag{27.23}$$

for all i, j, s and t with $i \neq j$. Note that the heteroskedasticity and serial correlation does not vary across cross-sections i.

Using the cross-section specific residual vectors, we may rewrite this assumption as,

$$E(\epsilon_i\epsilon_i'|X_i^*) = \Omega_T \tag{27.24}$$

for all i, where,

$$\Omega_T = \begin{pmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1T} \\ \sigma_{12} & \sigma_{22} & & \vdots \\ & & \ddots & \\ \sigma_{T1} & \cdots & & \sigma_{TT} \end{pmatrix} \tag{27.25}$$

We term this a *Period SUR* specification since it involves covariances across periods within a given cross-section, as in a seemingly unrelated regressions framework with period specific equations. In estimating a specification with Period SUR, we employ residuals obtained from first stage estimates to form an estimate of Ω_T. In the second stage, we perform feasible GLS.

See "Contemporaneous Covariances (Cross-section SUR)" on page 865 for related discussion.

Instrumental Variables

All of the pool specifications may be estimated using instrumental variables techniques. In general, the computation of the instrumental variables estimator is a straightforward extension of the standard OLS estimator. For example, in the simplest model, the OLS estimator may be written as:

$$\hat{\beta}_{OLS} = (\sum_i X_i'X_i)^{-1}(\sum_i X_i'Y_i) \tag{27.26}$$

while the corresponding IV estimator is given by:

$$\hat{\beta}_{IV} = (\sum_i X_i'P_{Z_i}X_i)^{-1}(\sum_i X_i'P_{Z_i}Y_i) \tag{27.27}$$

where $P_{Z_i} = (Z_i(Z_i'Z_i)^{-1}Z_i')$ is the orthogonal projection matrix onto the Z_i.

There are, however, additional complexities introduced by instruments that require some discussion.

Cross-section and Period Specific Instruments

As with the regressors, we may divide the instruments into three groups (common instruments Z_{0it}, cross-section specific instruments Z_{1it}, and period specific instruments Z_{2it}).

You should make certain that any exogenous variables in the regressor groups are included in the corresponding instrument groups, and be aware that each entry in the latter two groups generates multiple instruments.

Fixed Effects

If instrumental variables estimation is specified with fixed effects, EViews will automatically add to the instrument list any constants implied by the fixed effects so that the orthogonal projection is also applied to the instrument list. Thus, if Q is the fixed effects transformation operator, we have:

$$\begin{aligned}\hat{\beta}_{OLS} &= (\sum_i X_i'QX_i)^{-1}(\sum_i X_i'QY_i)\\ \hat{\beta}_{IV} &= (\sum_i X_iQP_{\tilde{Z}_i}QX_i)^{-1}(\sum_i X_i'QP_{\tilde{Z}_i}QY_i)\end{aligned} \tag{27.28}$$

where $\tilde{Z}_i = QZ_i$.

Random Effects and GLS

Similarly, for random effects and other GLS estimators, EViews applies the weighting to the instruments as well as the dependent variable and regressors in the model. For example, with data estimated using cross-sectional GLS, we have:

$$\begin{aligned}\hat{\beta}_{GLS} &= \left(\sum_i X_i'\hat{\Omega}_M^{-1}X_i\right)^{-1}\left(\sum_i X_i'\hat{\Omega}_M^{-1}Y_i\right)\\ \hat{\beta}_{GIV} &= \left(\sum_i X_i\hat{\Omega}_M^{-1/2}P_{Z_i*}\hat{\Omega}_M^{-1/2}X_i\right)^{-1}\left(\sum_i X_i'\hat{\Omega}_M^{-1/2}P_{Z_i*}\hat{\Omega}_M^{-1/2}Y_i\right)\end{aligned} \tag{27.29}$$

where $Z_i* = \hat{\Omega}_M^{-1/2}Z_i$.

In the context of random effects specifications, this approach to IV estimation is termed generalized two-stage least squares (G2SLS) method (see Baltagi (2001, pp. 111-115) for references and discussion). Note that in implementing the various random effects methods (Swamy-Arora, Wallace-Hussain, Wansbeek-Kapteyn), we have extended the existing results to derive the unbiased variance components estimators in the case of instrumental variables estimation.

More generally, the approach may simply be viewed as a special case of the Generalized Instrumental Variables (GIV) approach in which data and the instruments are both transformed using the estimated covariances. You should be aware that this has approach has the effect of altering the implied orthogonality conditions. See Wooldridge (2002) for discussion and comparison with a three-stage least squares approach in which the instruments are not transformed. See "GMM Details" on page 931 for an alternative approach.

AR Specifications

EViews estimates AR specifications by transforming the data to a nonlinear least squares specification, and jointly estimating the original and the AR coefficients.

This transformation approach raises questions as to what instruments to use in estimation. By default, EViews adds instruments corresponding to the lagged endogenous and lagged exogenous variables introduced into the specification by the transformation.

For example, in an AR(1) specification, we have the original specification,

$$Y_{it} = \alpha + X_{it}'\beta_i + \delta_i + \epsilon_{it} \tag{27.30}$$

and the transformed equation,

$$Y_{it} = \rho_{1i} Y_{it-1} + \alpha(1-\rho_{1i}) + (X_{it} - \rho_{1i} X_{it-1})'\beta_i + \delta_i(1-\rho_{1i}) + \eta_{it} \tag{27.31}$$

where Y_{it-1} and X_{it-1} are introduced by the transformation. EViews will, by default, add these to the previously specified list of instruments Z_{it}.

You may, however, instruct EViews not to add these additional instruments. Note, however, that the order condition for the transformed model is different than the order condition for the untransformed specification since we have introduced additional coefficients corresponding to the AR coefficients. If you elect not to add the additional instruments automatically, you should make certain that you have enough instruments to account for the additional terms.

Robust Coefficient Covariances

In this section, we describe the basic features of the various robust estimators, for clarity focusing on the simple cases where we compute robust covariances for models estimated by standard OLS without cross-section or period effects. The extensions to models estimated using instrumental variables, fixed or random effects, and GLS weighted least squares are straightforward.

White Robust Covariances

The White cross-section method is derived by treating the pool regression as a multivariate regression (with an equation for each cross-section), and computing White-type robust standard errors for the system of equations. We may write the coefficient covariance estimator as:

$$\left(\frac{N^*}{N^* - K^*}\right)\left(\sum_t X_t' X_t\right)^{-1}\left(\sum_t X_t' \hat{\epsilon}_t \hat{\epsilon}_t' X_t\right)\left(\sum_t X_t' X_t\right)^{-1} \tag{27.32}$$

where the leading term is a degrees of freedom adjustment depending on the total number of observations in the stacked data, N^* is the total number of stacked observations, and and K^*, the total number of estimated parameters.

This estimator is robust to cross-equation (contemporaneous) correlation as well as different error variances in each cross-section. Specifically, the unconditional contemporaneous variance matrix $E(\epsilon_t \epsilon_t') = \Omega_M$ is unrestricted, and the conditional variance matrix

$E(\epsilon_t \epsilon_t' | X_t^*)$ can depend on X_t^* in arbitrary, unknown fashion. See Wooldridge (2002, pp. 148-153) and Arellano (1987).

Alternatively, the White period method is robust to arbitrary serial correlation and time-varying variances in the disturbances. The coefficient covariances are calculated using:

$$\left(\frac{N^*}{N^* - K^*}\right)(\sum_i X_i' X_i)^{-1}(\sum_i X_i' \hat{\epsilon}_i \hat{\epsilon}_i' X_i)(\sum_i X_i' X_i)^{-1} \tag{27.33}$$

where, in contrast to Equation (27.32), the summations are taken over individuals and individual stacked data instead of periods.

The White period robust coefficient variance estimator is designed to accommodate arbitrary serial correlation and time-varying variances in the disturbances. The corresponding multivariate regression (with an equation for each period) allows the unconditional variance matrix $E(\epsilon_i \epsilon_i') = \Omega_T$ to be unrestricted, and the conditional variance matrix $E(\epsilon_i \epsilon_i' | X_i^*)$ may depend on X_i^* in general fashion.

In contrast, the White (diagonal) method is robust to observation specific heteroskedasticity in the disturbances, but not to correlation between residuals for different observations. The coefficient asymptotic variance is estimated as:

$$\left(\frac{N^*}{N^* - K^*}\right)(\sum_{i,t} X_{it}' X_{it})^{-1}\left(\sum_{i,t} \hat{\epsilon}_{it}^2 X_{it}' X_{it}\right)(\sum_{i,t} X_{it}' X_{it})^{-1} \tag{27.34}$$

This method allows the unconditional variance matrix $E(\epsilon\epsilon') = \Lambda$ to be an unrestricted diagonal matrix, and the conditional variances $E(\epsilon_{it}^2 | X_{it}^*)$ to depend on X_{it}^* in general fashion. Note that this method is both more general and more restrictive than the previous approaches. It is more general in that observations in the same cross-section or period may have different variances; it is more restrictive in that all off-diagonal variances are restricted to be zero.

EViews also allows you to compute non degree of freedom corrected versions of all of the robust coefficient covariance estimators. In these cases, the leading ratio term in the expressions above is dropped from the calculation. While this has no effect on the asymptotic validity of the estimates, it has the practical effect of lowering all of your standard error estimates.

PCSE Robust Covariances

The remaining methods are variants of the first two White statistics in which residuals are replaced by moment estimators for the unconditional variances. These methods, which are variants of the so-called *Panel Corrected Standard Error* (PCSE) methodology (Beck and Katz, 1995), are robust to unrestricted unconditional variances Ω_M and Ω_T, but place additional restrictions on the conditional variance matrices. A sufficient (though not necessary) condition is that the conditional and unconditional variances are the same.

For example, the Cross-section SUR (PCSE) method replaces the outer product of the cross-section residuals in Equation (27.32) with an estimate of the cross-section residual (contemporaneous) covariance matrix Ω_M:

$$\left(\frac{N^*}{N^* - K^*}\right)\left(\sum_t X_t' X_t\right)^{-1}\left(\sum_t X_t' \hat{\Omega}_M X_t\right)\left(\sum_t X_t' X_t\right)^{-1} \tag{27.35}$$

Analogously, the Period SUR (PCSE) replaces the outer product of the period residuals in Equation (27.33) with an estimate of the period covariance Ω_T:

$$\left(\frac{N^*}{N^* - K^*}\right)\left(\sum_i X_i' X_i\right)^{-1}\left(\sum_i X_i' \hat{\Omega}_T X_i\right)\left(\sum_i X_i' X_i\right)^{-1} \tag{27.36}$$

The two diagonal forms of these estimators, Cross-section weights (PCSE), and Period weights (PCSE), use only the diagonal elements of the relevant $\hat{\Omega}_M$ and $\hat{\Omega}_T$. These covariance estimators are robust to heteroskedasticity across cross-sections or periods, respectively, but not to general correlation of residuals.

The non degree of freedom corrected versions of these estimators remove the leading term involving the number of observations and number of coefficients.

Chapter 28. Working with Panel Data

EViews provides you with specialized tools for working with stacked data that have a panel structure. You may have, for example, data for various individuals or countries that are stacked one on top of another.

The first step in working with stacked panel data is to describe the panel structure of your data: we term this step *structuring the workfile*. Once your workfile is structured as a panel workfile, you may take advantage of the EViews tools for working with panel data, and for estimating equation specifications using the panel structure.

The following discussion assumes that you have an understanding of the basics of panel data. "Panel Data" beginning on page 210 provides background on the characteristics of panel structured data.

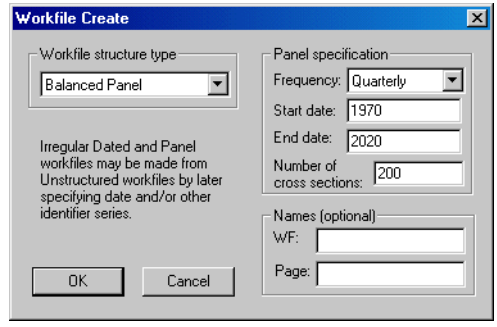
We first review briefly the process of applying a panel structure to a workfile. The remainder of the discussion in this chapter focuses on the basics working with data in a panel workfile. Chapter 29, "Panel Estimation", on page 901 outlines the features of equation estimation in a panel workfile.

Structuring a Panel Workfile

The first step in panel data analysis is to define the panel structure of your data. By defining a panel structure for your data, you perform the dual tasks of identifying the cross-section associated with each observation in your stacked data, and of defining the way that lags and leads operate in your workfile.

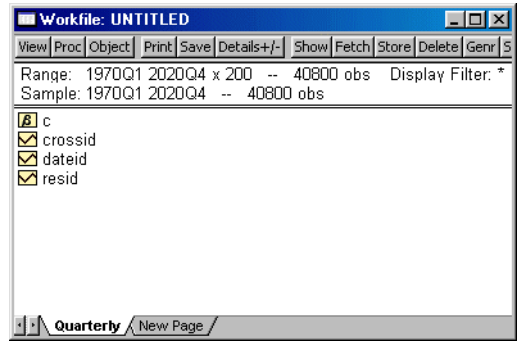
While the procedures for structuring a panel workfile outlined below are described in greater detail elsewhere, an abbreviated review may prove useful (for additional detail, see "Describing a Balanced Panel Workfile" on page 53, "Dated Panels" on page 223, and "Undated Panels" on page 228).

There are two basic ways to create a panel structured workfile. First, you may create a *new* workfile that has a simple balanced panel structure. Simply select **File/New/Workfile...** from the main EViews menu to open the **Workfile Create** dialog. Next, select **Balanced Panel** from the **Workfile structure type** combo box, and fill out the dialog as desired.



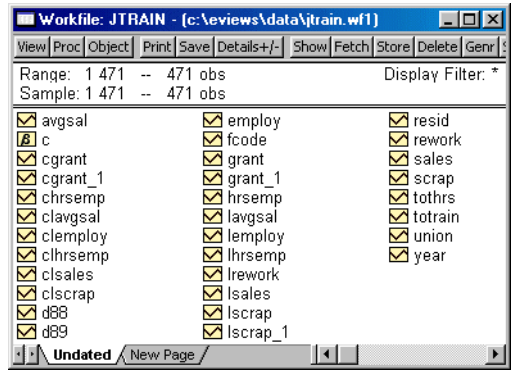
Here, we create a balanced quarterly panel (ranging from 1970Q1 to 2020Q4) with 200 cross-sections.

When you click on **OK**, EViews will create an appropriately structured workfile with 40,800 observations (51 years, 4 quarters, 200 cross-sections). You may then enter or import the data into the workfile.



More commonly, you will use the second method of structuring a panel workfile, in which you first read stacked data into an unstructured workfile, and then apply a structure to the workfile. While there are a number of issues involved with this operation, let us consider a simple, illustrative example of the basic method.

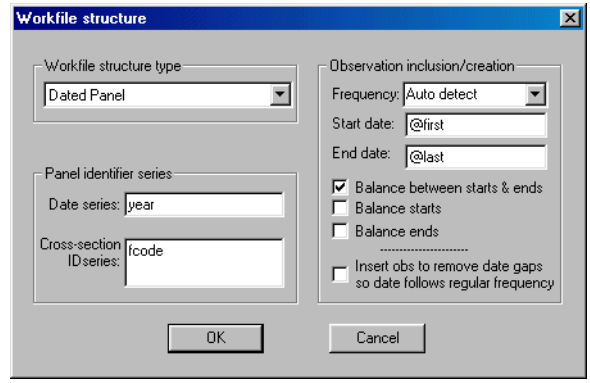
Suppose that we have data for the job training example considered by Wooldridge (2002), using data from Holzer, *et al.* (1993).



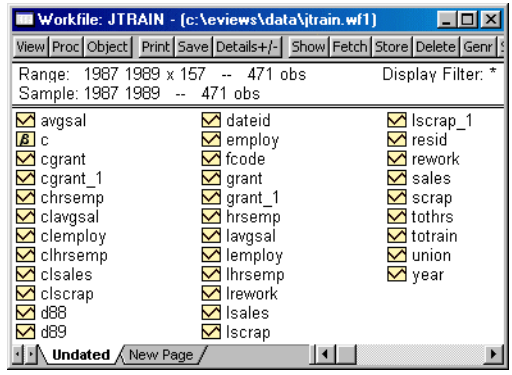
These data form a balanced panel of 3 annual observations on 157 firms. The data are first read into a 471 observation, unstructured EViews workfile. The values of the series YEAR and FCODE may be used to identify the date and cross-section, respectively, for each observation.

To apply a panel structure to this workfile, simply double click on the "Range:" line at the top of the workfile window, or select **Proc/Structure/Resize Current Page...** to open the **Workfile structure** dialog. Select **Dated Panel** as our **Workfile structure type**.

Next, enter YEAR as the **Date series** and FCODE as the **Cross-section ID series**. Since our data form a simple balanced dated panel, we need not concern ourselves with the remaining settings, so we may simply click on **OK**.



EViews will analyze the data in the specified **Date series** and **Cross-section ID series** to determine the appropriate structure for the workfile. The data in the workfile will be sorted by cross-section ID series, and then by date, and the panel structure will be applied to the workfile.



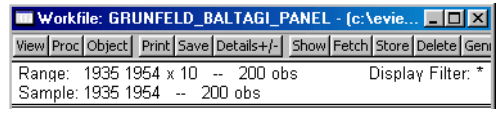
Panel Workfile Display

The two most prominent visual changes in a panel structured workfile are the change in the range and sample information display at the top of the workfile window, and the change in the labels used to identify individual observations.

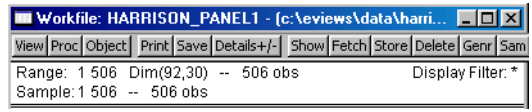
Range and Sample

The first visual change in a panel structured workfile is in the **Range** and **Sample** descriptions at the top of workfile window.

For a dated panel workfile, EViews will list both the earliest and latest observed dates, the number of cross-sections, and the total number of unique observations.


Workfile: GRUNFELD_BALTAGI_PANEL - (c:\evie...
View Proc Object Print Save Details+/- Show Fetch Store Delete Geni
Range: 1935 1954 x 10 -- 200 obs
Display Filter: *
Sample: 1935 1954 -- 200 obs

Here we see the top portion of an annual workfile with observations from 1935 to 1954 for 10 cross-sections. Note that workfile sample is described using the earliest and latest observed annual frequency dates ("1935 1954").


Workfile: HARRISON_PANEL1 - (c:\eviews\data\harri...
View Proc Object Print Save Details+/- Show Fetch Store Delete Genr Sam
Range: 1 506 Dim(92,30) -- 506 obs Display Filter: *
Sample: 1 506 -- 506 obs

In contrast, an undated panel workfile will display an observation range of 1 to the total number of observations. The panel dimension statement will indicate the largest number of observations in a cross-section and the number of cross-sections. Here, we have 92 cross-sections containing up to 30 observations, for a total of 506 observations. Note that the workfile sample is described using the raw observation numbers ("1 506") since there is no notion of a date pair in undated panels.

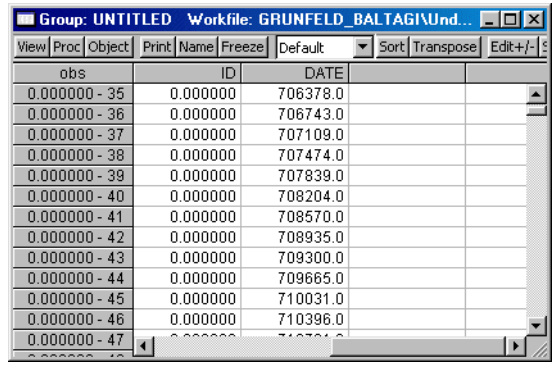
You may, at any time, click on the **Range** display line or select **Proc/WF Structure & Range...** to bring up the **Workfile Structure** dialog so that you may modify or remove your panel structure.

Observation Labels

The left-hand side of every workfile contains observation labels that identify each observation. In a simple unstructured workfile, these labels are simply the integers from 1 to the total number of observations in the workfile. For dated, non-panel workfiles, these labels are representations of the unique dates associated with each observation. For example, in an annual workfile ranging from 1935 to 1950, the observation labels are of the form "1935", "1936", etc.

In contrast, the observation labels in a panel workfile must reflect the fact that observations possess both cross-section and within-cross-section identifiers. Accordingly, EViews will form observation identifiers using both the cross-section and the cell ID values.

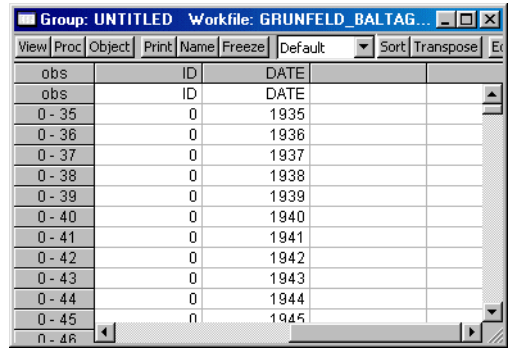
Here, for example, we see the labels for a panel annual workfile structured using ID as the cross-section identifier, and DATE as the cell ID (we have widened the observation column so that the full observation identifier is visible). The labels have been formed by concatenating the ID value, a hyphen, and a two-digit representation of the date.



Note that EViews generally uses the display formats for the cross-section and cell ID series when forming the observation label. For example, since ID is displayed using 7 fixed char-

acters, the observation labels include 7 fixed characters for the ID value. A notable exception to this rule is seen in the DATE portion of the observation label. Since we have specified a dated annual panel format, EViews uses this information to shorten the date portion of the observation label by using a 2-digit year identifier.

We may improve the appearance of the observation labels to changing the spreadsheet display settings in the ID series. Simply open the ID series window, click on the **Properties** button and select the **Display** tab, then change the display to show significant digits. Similarly, we may open the DATE series and specify display properties to show the dates using the **Current workfile** format.


Group: UNTITLED Workfile: GRUNFELD_BALTAG...
View Proc Object Print Name Freeze Default Sort Transpose
obs ID DATE
obs ID DATE
0 - 35 0 1935
0 - 36 0 1936
0 - 37 0 1937
0 - 38 0 1938
0 - 39 0 1939
0 - 40 0 1940
0 - 41 0 1941
0 - 42 0 1942
0 - 43 0 1943
0 - 44 0 1944
0 - 45 0 1945
0 - 46

Here, we show the observation labels along with formatted values for ID and DATE. Note again that the observation labels do not use the formatting for DATE, but instead use formatting based on the workfile date structure.

Panel Workfile Information

When working with panel data, it is important to keep the basic structure of your workfile in mind at all times. EViews provides you with tools to access information about the structure of your workfile.

Workfile Structure

First, the workfile statistics view provides a convenient place for you to examine the structure of your panel workfile. Simply click on **View/Statistics** from the main menu to display a summary of the structure and contents of your workfile.

Workfile Statistics
Date: 01/13/04 Time: 08:29
Name: GRUNFELD_BALTAGI_PANEL
Number of pages: 3

Page: panel
Workfile structure: Panel - Annual
Indices: ID x DATE
Panel dimension: 10 x 20
Range: 1935 1954 x 10 -- 200 obs

Object	Count	Data Points
series	7	1400
alpha	1	200
coef	1	751
equation	10	
group	2	
Total	21	2351

The top portion of the display for our first example workfile is depicted above. The statistics view identifies the page as an annual panel workfile that is structured using the identifiers ID and DATE. There are 10 cross-sections with 20 observations each, for years ranging from 1935 to 1954. For unbalanced data, the number of observations per cross-section reported will be the largest number observed across the cross-sections.

To return the display to the original workfile directory, select **View/Workfile Directory** from the main workfile menu.

Identifier Indices

EViews provides series expressions and functions that provide information about the cross-section, cell, and observation IDs associated with each observation in a panel workfile.

Cross-section Index

The series expression @CROSSID provides index identifiers for each observation corresponding to the cross-section to which the observation belongs. If, for example, there are 8 observations with cross-section identifier series values (in order), “B”, “A”, “A”, “A”, “B”, “A”, “A”, and “B”, the command:

```
series cxid = @crossid
```

assigns a group identifier value of 1 or 2 to each observation in the workfile. Since the panel workfile is sorted by the cross-section ID values, observations with the identifier value “A” will be assigned a CXID value of 1, while “B” will be assigned 2.

A one-way tabulation of the CXID series shows the number of observations in each cross-section or group:

Tabulation of CXID
Date: 02/04/04 Time: 09:08
Sample: 1 8
Included observations: 8
Number of categories: 2

Value	Count	Percent	Cumulative Count	Cumulative Percent
1	5	62.50	5	62.50
2	3	37.50	8	100.00
Total	8	100.00	8	100.00

Cell Index

Similarly, @CELLID may be used to obtain integers uniquely indexing cell IDs. @CELLID numbers observations using an index corresponding to the ordered unique values of the cell or date ID values. Note that since the indexing uses all unique values of the cell or date ID series, the observations within a cross-section may be indexed non-sequentially.

Suppose, for example, we have a panel workfile with two cross-sections. There are 5 observations in the cross-section "A" with cell ID values "1991", "1992", "1993", "1994", and "1999", and 3 observations in the cross-section "B" with cell ID values "1993", "1996", "1998". There are 7 unique cell ID values ("1991", "1992", "1993", "1994", "1996", "1998", "1999") in the workfile.

The series assignment

```
series cellid = @cellid
```

will assign to the "A" observations in CELLID the values "1991", "1992", "1993", "1994", "1997", and to the "B" observations the values "1993", "1995", and "1996".

A one-way tabulation of the CELLID series provides you with information about the number of observations with each index value:

Tabulation of CELLID
Date: 02/04/04 Time: 09:11
Sample: 1 8
Included observations: 8
Number of categories: 7

Value	Count	Percent	Cumulative Count	Cumulative Percent
1	1	12.50	1	12.50
2	1	12.50	2	25.00
3	2	25.00	4	50.00
4	1	12.50	5	62.50
5	1	12.50	6	75.00
6	1	12.50	7	87.50
7	1	12.50	8	100.00
Total	8	100.00	8	100.00

Within Cross-section Observation Index

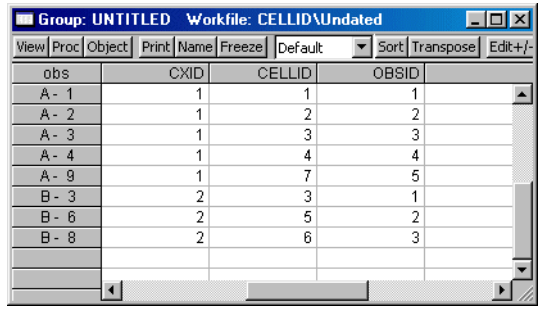
Alternately, @OBSID returns an integer uniquely indexing observations within a cross-section. The observations will be numbered sequentially from 1 through the number of observations in the corresponding cross-section. In the example above, with two cross-section groups "A" and "B" containing 5 and 3 observations, respectively, the command:

```
series withinid = @obsid
```

would number the 5 observations in cross-section "A" from 1 through 5, and the 3 observations in group "B" from 1 through 3.

Bear in mind that while @CELLID uses information about all of the ID values in creating its index, @OBSID only uses the ordered observations within a cross-section in forming the index. As a result, the only similarity between observations that share an @OBSID value is their ordering within the cross-section. In contrast, observations that share a @CELLID value also share values for the underlying cell ID.

It is worth noting that if a panel workfile is balanced so that each cross-section has the same cell ID values, @OBSID and @CELLID yield identical results.



obs	CXID	CELLID	OBSID
A- 1	1	1	1
A- 2	1	2	2
A- 3	1	3	3
A- 4	1	4	4
A- 9	1	7	5
B- 3	2	3	1
B- 6	2	5	2
B- 8	2	6	3

Workfile Observation Index

In rare cases, you may wish to enumerate the observations beginning at the first observation in the first cross-section and ending at the last observation in the last cross-section.

```
series _id = @obsnum
```

The @OBSNUM keyword allows you to number the observations in the workfile in sequential order from 1 to the total number of observations.

Working with Panel Data

For the most part, you will find working with data in a panel workfile to be identical to working with data in any other workfile. There are, however, some differences in behavior that require discussion. In addition, we describe useful approaches to working with panel data using standard, non panel-specific tools.

Lags and Leads

For the most part, expressions involving lags and leads should operate as expected (see "Lags, Leads, and Panel Structured Data" on page 212 for a full discussion). In particular note that lags and leads do not cross group boundaries so that they will never involve data from a different cross-section (*i.e.*, lags of the first observation in a cross-section are always NAs, as are leads of the last observation in a cross-section).

Since EViews automatically sorts your data by cross-section and cell/date ID, observations in a panel dataset are always stacked by cross-section, with the cell IDs sorted within each cross-section. Accordingly, lags and leads within a cross-section are defined over the sorted values of the cell ID. Lags of an observation are always associated with lower value of the cell ID, and leads always involve a higher value (the first lag observation has the next lowest cell ID value and the first lead has the next highest value).

Lags and leads are specified in the usual fashion, using an offset in parentheses. To assign the sum of the first lag of Y and the second lead of X to the series Z, you may use the command:

```
series z = y(-1) + x(2)
```

Similarly, you may use lags to obtain the name of the previous child in household cross-sections. The command:

```
alpha older = childname(-1)
```

assigns to the alpha series OLDER the name of the preceding observation. Note that since lags never cross over cross-section boundaries, the first value of OLDER in a household will be missing.

Panel Samples

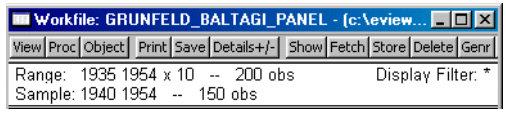
The description of the current workfile sample in the workfile window provides an obvious indication that samples for dated and undated workfiles are specified in different ways.

Dated Panel Samples

For dated workfiles, you may specify panel samples using date pairs to define the earliest and latest dates to be included. For example, in our dated panel example from above, if we issue the sample statement:

```
smpl 1940 1954
```

EViews will exclude all observations that are dated from 1935 through 1939. We see that the new sample has eliminated observations for those dates from each cross-section.


Workfile: GRUNFELD_BALTAGI_PANEL - (c:\eview...
View Proc Object Print Save Details+/- Show Fetch Store Delete Genr
Range: 1935 1954 x 10 -- 200 obs
Display Filter: *
Sample: 1940 1954 -- 150 obs

As in non-panel workfiles, you may combine the date specification with additional "if" conditions to exclude additional observations. For example:

```
smpl 1940 1945 1950 1954 if i>50
```

uses any panel observations that are dated from 1940 to 1945 or 1950 to 1954 that have values of the series I that are greater than 50.

Additionally, you may use special keywords to refer to the first and last observations for cross-sections. For dated panels, the sample keywords @FIRST and @LAST refer to the set of first and last observations for each cross-section. For example, you may specify the sample:

```
smpl @first 2000
```

to use data from the first observation in each cross-section and observations up through the end of the year 2000. Likewise, the two sample statements:

```
smpl @first @first+5
smpl @last-5 @last
```

use (at most) the first five and the last five observations in each cross-section, respectively.

Note that the included observations for each cross-section may begin at a different date, and that:

```
smpl @all
smpl @first @last
```

are equivalent.

The sample statement keywords @FIRSTMIN and @LASTMAX are used to refer to the earliest of the start and latest of the end dates observed over all cross-sections, so that the sample:

```
smpl @firstmin @firstmin+20
```

sets the start date to the earliest observed date, and includes the next 20 observations in each cross-section. The command:

```
smpl @lastmax-20 @lastmax
```

includes the last observed date, and the previous 20 observations in each cross-section.

Similarly, you may use the keywords @FIRSTMAX and @LASTMIN to refer to the latest of the cross-section start dates, and earliest of the end dates. For example, with *regular* annual data that begin and end at different dates, you may balance the starts and ends of your data using the statement:

```
smpl @firstmax @lastmin
```

which sets the sample to begin at the latest observed start date, and to end at the earliest observed end date.

The special keywords are perhaps most usefully combined with observation offsets. By adding plus and minus terms to the keywords, you may adjust the sample by dropping or adding observations within each cross-section. For example, to drop the first observation from each cross-section, you may use the sample statement:

```
smpl @first+1 @last
```

The following commands generate a series containing cumulative sums of the series X for each cross-section:

```
smpl @first @first
series xsum = x
smpl @first+1 @last
xsum = xsum(-1) + x
```

The first two commands initialize the cumulative sum for the first observation in each cross-section. The last two commands accumulate the sum of values of X over the remaining observations.

Similarly, if you wish to estimate your equation on a subsample of data and then perform cross-validation on the last 20 observations in each cross-section, you may use the sample defined by,

```
smpl @first @last-20
```

to perform your estimation, and the sample,

```
smpl @last-19 @last
```

to perform your forecast evaluation.

Note that the processing of sample offsets for each cross-section follows the same rules as for non-panel workfiles "Sample Offsets" on page 99.

Undated Panel Samples

For undated workfiles, you must specify the sample range pairs using observation numbers defined over the entire workfile. For example, in our undated 506 observation panel example, you may issue the sample statement:

```
smpl 10 500
```

to drop the first 9 and the last 6 observations in the workfile from the current sample.

One consequence of the use of observation pairs in undated panels is that the keywords "@FIRST", "@FIRSTMIN", and "@FIRSTMAX" all refer to observation 1, and "@LAST", "@LASTMIN", and "@LASTMAX", refer to the last observation in the workfile. Thus, in our example, the command:

```
smpl @first+9 @lastmax-6
```

will also drop the first 9 and the last 6 observations in the workfile from the current sample.

Undated panel sample restrictions of this form are not particularly interesting since they require detailed knowledge of the pattern of observation numbers across those cross-sections. Accordingly, most sample statements in undated workfiles will employ "IF conditions" in place of range pairs.

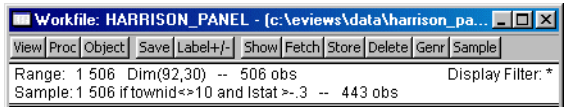
For example, the sample statement,

```
smpl if townid<>10 and lstat >-.3
```

is equivalent to either of the commands,

```
smpl @all if townid<>10 and lstat >-.3
smpl 1 506 if townid<>10 and lstat >-.3
```

and selects all observations with TOWNID values not equal to 10, and LSTAT values greater than -0.3.


Workfile: HARRISON_PANEL - (c:\eviews\data\harrison_pa...
View Proc Object Save Label+/- Show Fetch Store Delete Genr Sample
Range: 1 506 Dim(92,30) -- 506 obs Display Filter: *
Sample: 1 506 if townid<>10 and lstat >-.3 -- 443 obs

You may combine the sample "IF conditions" with the special functions that return information about the observations in the panel. For example, we may use the "@OBSID" workfile function to identify each observation in a cross-section, so that:

```
smpl if @obsid>1
```

drops the first observation for each cross-section.

Alternately, to drop the last observation in each cross-section, you may use:

```
smpl if @obsid < @maxsby(townid, townid, "@all")
```

The @MAXSBY function returns the number of non-NA observations for each TOWNID value. Note that we employ the "@ALL" sample to ensure that we compute the @MAXSBY over the entire workfile sample.

Trends

EViews provides several functions that may be used to construct a time trend in your panel structured workfile. A trend in a panel workfile has the property that the values are initialized at the start of a cross-section, increase for successive observations in the specific cross-section, and are reset at the start of the next cross section.

You may use the following to construct your time trend:

- The @OBSID function may be used to return the simplest notion of a trend in which the values for each cross-section begin at one and increase by one for successive observations in the cross-section.
- The @TRENDC function computes trends in which values for observations with the earliest observed date are normalized to zero, and values for successive observations are incremented based on the calendar associated with the workfile frequency.
- The @CELLID and @TREND functions return time trends in which the values increase based on a calender defined by the observed dates in the workfile.

See also "Trend Functions" on page 592 and "Panel Trend Functions" on page 593 for discussion.

By-Group Statistics

The "by-group" statistical functions ("By-Group Statistics" on page 579) may be used to compute the value of a statistic for observations in a subgroup, and to assign the computed value to individual observations.

While not strictly panel functions, these tools deserve a place in the current discussion since they are well suited for working with panel data. To use the by-group statistical functions in a panel context, you need only specify the group ID series as the classifier series in the function.

Suppose, for example, that we have the undated panel structured workfile with the group ID series TOWNID, and that you wish to assign to each observation in the workfile the mean value of LSTAT in the corresponding town. You may perform the series assignment using the command,

```
series meanlstat = @meansby(lstat, townid, "@all")
```

or equivalently,

```
series meanlstat = @meansby(lstat, @crossid, "@all")
```

to assign the desired values. EViews will compute the mean value of LSTAT for observations with each TOWNID (or equivalently @CROSSID, since the workfile is structured using TOWNID) value, and will match merge these values to the corresponding observations.

Likewise, we may use the by-group statistics functions to compute the variance of LSTAT or the number of non-NA values for LSTAT for each subgroup using the assignment statements:

```
series varlstat = @varsby(lstat, townid, "@all")
series nalstat = @nasby(lstat, @crossid, "@all")
```

To compute the statistic over subsamples of the workfile data, simply include a sample string or object as an argument to the by-group statistic, or set the workfile sample prior to issuing the command,

```
smpl @all if zn=0
series meanlstat1 = @meansby(lstat, @cellid)
```

is equivalent to,

```
smpl @all
series meanlstat2 = @meansby(lstat, @cellid, "@all if zn=0")
```

In the former example, the by-group function uses the workfile sample to compute the statistic for each cell ID value, while in the latter, the optional argument explicitly overrides the workfile sample.

One important application of by-group statistics is to compute the "within" deviations for a series by subtracting off panel group means or medians. The following lines:

```
smpl @all
series withinlstat1 = lstat - @meansby(lstat, townid)
series withinlstat2 = lstat - @mediansby(lstat, townid)
```

compute deviations from the TOWNID specific means and medians. In this example, we omit the optional sample argument from the by-group statistics functions since the workfile sample is previously set to use all observations.

Combined with standard EViews tools, the by-group statistics allow you to perform quite complex calculations with little effort. For example, the panel "within" standard deviation for LSTAT may be computed from the single command:

```
scalar within_std = @stdev(lstat - @meansby(lstat, townid,
   "@all"))
```

while the "between" standard deviation may be calculated from

```
smpl if @obsid = 1
scalar between_std = @stdev(@meansby(lstat, @crossid, "@all"))
```

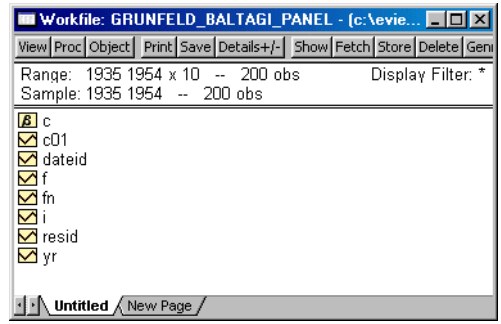
The first line sets the sample to the first observation in each cross-section. The second line calculates the standard deviation of the group means using the single cross-sectional observations. Note that the group means are calculated over the entire sample. An alternative approach to performing this calculation is described in the next section.

Cross-section and Period Summaries

One of the most important tasks in working with panel data is to compute and save summary data, for example, computing means of a series by cross-section or period. In "By-Group Statistics" on page 886, we outlined tools for computing by-group statistics using the cross-section ID and match merging them back into the original panel workfile page.

Additional tools are available for displaying tables summarizing the by-group statistics or for saving these statistics into new workfile pages.

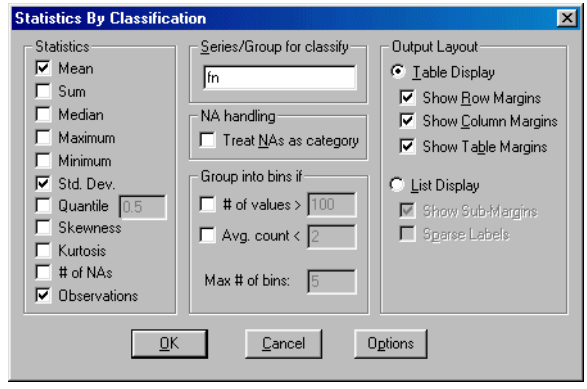
In illustrating these tools, we will work with the familiar Grunfeld data containing data on R&D expenditure and other economic measures for 10 firms for the years 1935 to 1954. These 200 observations form a balanced annual workfile that is structured using the firm number FN as the cross-section ID series, and the date series DATEID to identify the year.



Viewing Summaries

The easiest way to compute by-group statistics is to use the standard by-group statistics view of a series. Simply open the series window and select **View/Descriptive Statistics/Stats by Classification...** to open the **Statistics by Classification** dialog.

First, you should enter the classifier series in the **Series/Group to classify** edit field. Here, we use FN, so that EViews will compute means, standard deviations, and number of observations for each cross-section in the panel workfile. Note that we have unchecked the **Group into bins** options so that EViews will not combine periods. The result of this computation for the series F is given by:



Descriptive Statistics for F
Categorized by values of FN
Date: 02/04/04 Time: 15:26
Sample (adjusted): 1935 1954
Included observations: 200 after adjustments

FN	Mean	Std. Dev.	Obs.
1	4333.845	904.3048	20
2	1971.825	301.0879	20
3	1941.325	413.8433	20
4	693.2100	160.5993	20
5	231.4700	73.84083	20
6	419.8650	217.0098	20
7	149.7900	32.92756	20
8	670.9100	222.3919	20
9	333.6500	77.25478	20
10	70.92100	9.272833	20
All	1081.681	1314.470	200

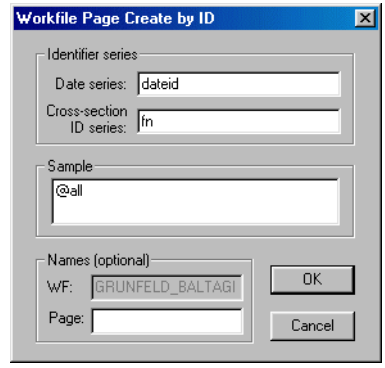
Alternately, to compute statistics for each period in the panel, you should enter "DATEID" instead of "FN" as the classifier series.

Saving Summaries

Alternately, you may wish to compute the by-group panel statistics and save them in their own workfile page. The standard EViews tools for working with workfiles and creating series links make this task virtually effortless.

Creating Pages for Summaries

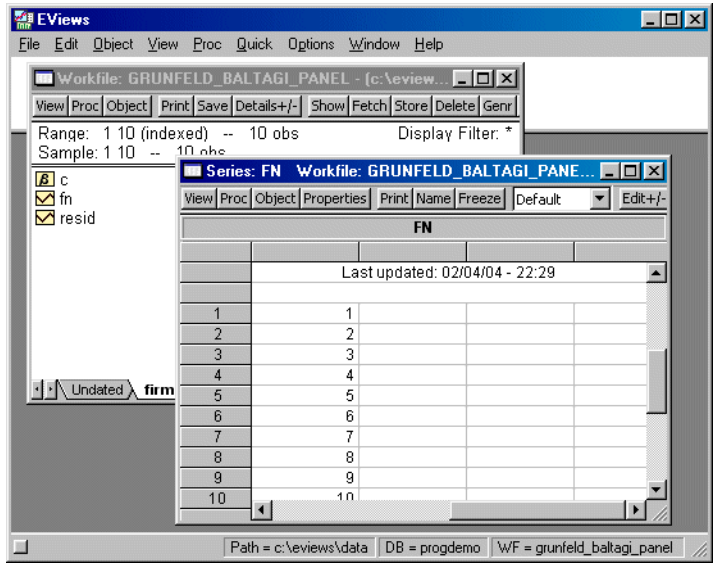
Since we will be computing both by-firm and by-period descriptive statistics, the first step is to create workfile pages to hold the results from our two sets of calculations. The firm page will contain observations corresponding to the unique values of the firm identifier found in the panel page; the annual page will contain observations corresponding to the observed years.



To create a page for the firm data, click on the **New Page** tab in the workfile window, and select **Specify by Identifier series...**. EViews opens the **Workfile Page Create by ID** dialog, with the identifiers pre-filled with the series used in the panel workfile structure—the **Date series** field contains the name of the series used to identify dates in the panel, while the **Cross-section ID** series field contains the name of the series used to identify firms.

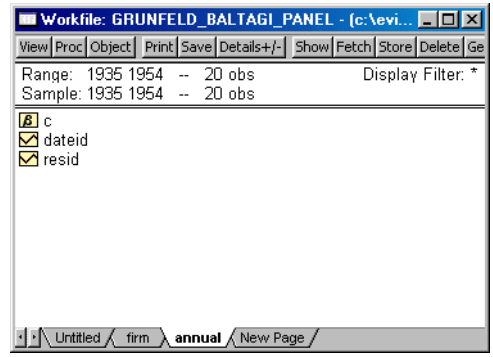
To create a new workfile using only the values in the FN series, you should delete the **Date series** specification "DATEID" from the dialog. Next, provide a name for the new page by entering "firm" in the **Page** edit field. Now click on **OK**.

EViews will examine the FN series to find its unique values, and will create and structure a workfile page to hold those values.



Here, we see the newly created FIRM page and newly created FN series containing the unique values from FN in the other page. Note that the new page is structured as an **Undated with ID series** page, using the new FN series.

Repeating this process using the DATEID series will create an annual page. First click on the original panel page to make it active, then select **New Page/Specify by Identifier series...** to bring up the previous dialog. Delete the **Cross-section ID series** specification "FN" from the dialog, provide a name for the new page by entering "annual" in the **Page** edit field, and click on **OK**. EViews creates the third page, a regular frequency annual page dated 1935 to 1954.

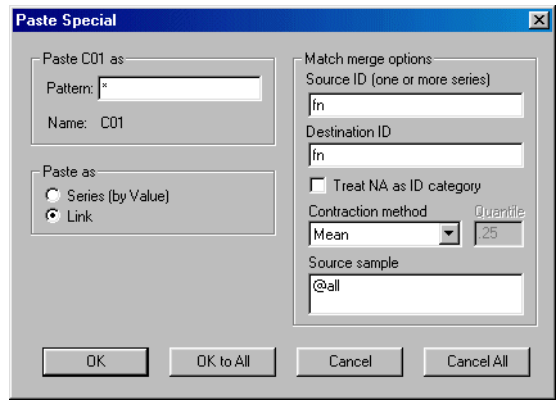

Workfile: GRUNFELD_BALTAGI_PANEL - (c:\evi...
View Proc Object Print Save Details+/- Show Fetch Store Delete Ge
Range: 1935 1954 -- 20 obs
Display Filter: *
Sample: 1935 1954 -- 20 obs
c
dateid
resid
Untitled firm annual New Page

Computing Summaries using Links

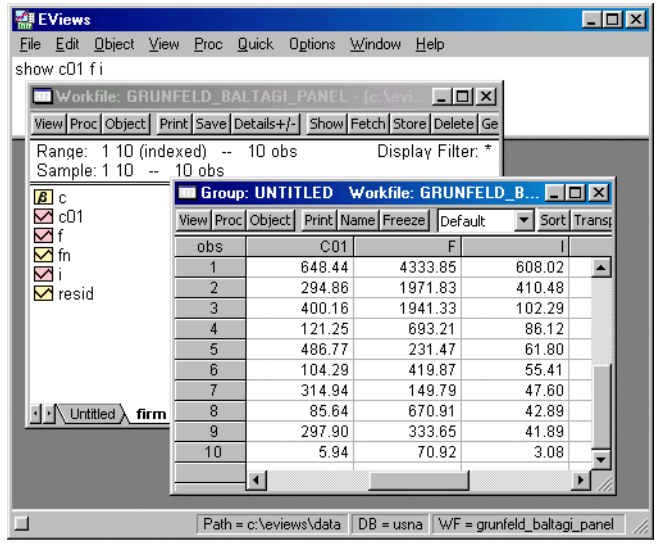
Once the firm and annual pages have been created, it is a simple task to create by-group summaries of the panel data using series links. While links are described elsewhere in greater depth (Chapter 8, "Series Links", on page 177), we provide a brief description of their use in a panel data context.

To create links containing the desired summaries, first click on the original panel page tab to make it active, select one or more series of interest, then right mouse click and select **Copy**. Next, click on either the firm or the annual page, right mouse click, and select **Paste Special...**. EViews will open the **Link Dialog**, prompting you to specify a method for summarizing the data.

Suppose, for example, that you select and copy the C01, F, and I series from the panel page and then select **Paste Special...** in the firm page. EViews analyzes the two pages and prefills the **Source ID** and **Destination ID** series with two FN cross-section ID series.



You may provide a different pattern to be used in naming the link series, a contraction method, and a sample over which the contraction should be calculated. Here, we create new series with the same names as the originals, computing means over the entire sample in the panel page. Click on **OK to All** to link all three series into the firm page, yielding:



You may compute other summary statistics by repeating the copy-and-paste-special procedure using alternate contraction methods. For example, selecting the **Standard Deviation** contraction computes the standard deviation for each cross-section and specified series and uses the linking to merge the results into the firm page. Saving them using the pattern "*SD" will create links named "C01SD", "FSD", and "ISD".

Likewise, to compute summary statistics across cross-sections for each year, first create an annual page using **New Page/Specify by Identifier series...,** then paste-special the panel page series as links in the annual page.

Merging Data into the Panel

To merge data into the panel, simply create links from other pages into the panel page. Linking from the annual page into the panel page will repeat observations for each year across firms. Similarly, linking from the cross-section firm page to the panel page will repeat observations for each firm across all years.

In our example, we may link the FSD link from the firm page back into the panel page. Select FSD, switch to the panel page, and paste-special. Click **OK** to accept the defaults in the **Paste Special** dialog.

EViews match merges the data from the firm page to the panel page, matching FN values. Since the merge is from one-to-many, EViews simply repeats the values of FSD in the panel page.

Basic Panel Analysis

EViews provides various degrees of support for the analysis of data in panel structured workfiles.

There is a small number of panel-specific analyses that are provided for data in panel structured workfiles. You may use EViews special tools for graphing dated panel data, perform unit root tests, or estimate various panel equation specifications.

Alternately, you may apply EViews standard tools for by-group analysis to the stacked data. These tools do not use the panel structure of the workfile, *per se*, but used appropriately, the by-group tools will allow you to perform various forms of panel analysis.

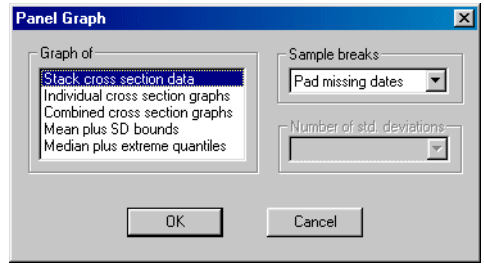
In most other cases, EViews will simply treat panel data as a set of stacked observations. The resulting stacked analysis correctly handles leads and lags in the panel structure, but does not otherwise use the cross-section and cell or period identifiers in the analysis.

Panel-Specific Analysis

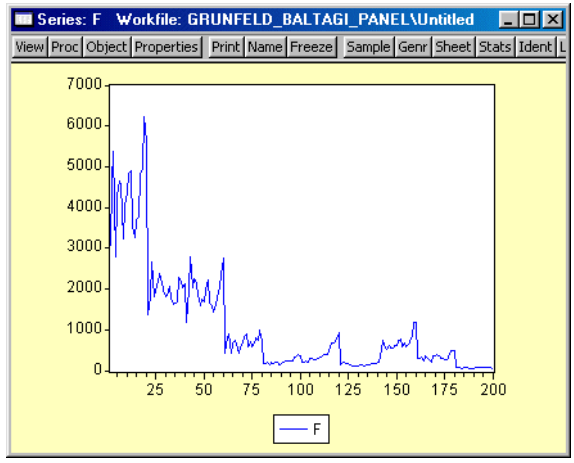
Time Series Graphs

EViews provides tools for displaying time series graphs with panel data. You may use these tools to display a graph of the stacked data, individual or combined graphs for each cross-section, or a time series graph of summary statistics for each period.

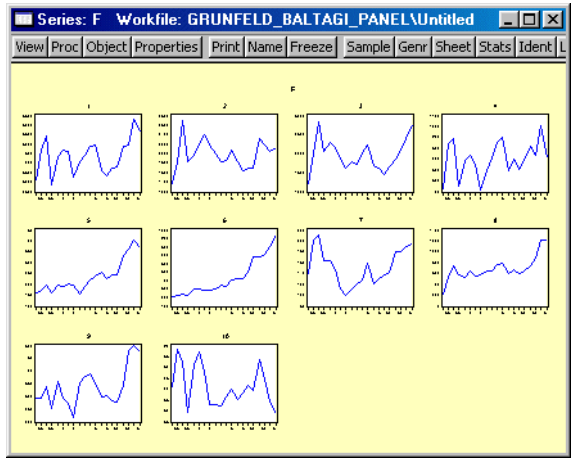
To display panel graphs for a series in a dated workfile, open the series window, click on **View/Graph**, and then select one of the graph types, for example **Line**. EViews will display a dialog offering you a variety of choices for how you wish to display the data.



If you select **Stack cross-section data**, EViews will display a single graph of the stacked data, numbered from 1 to the total number of observations.

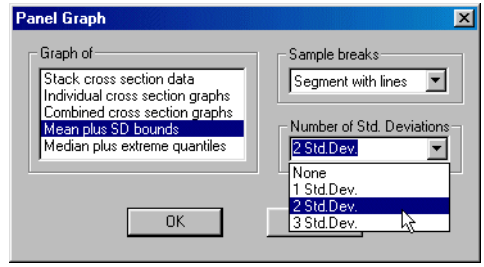


Alternately, selecting **Individual cross-section graphs** displays separate time series graphs for each cross-section, while **Combined cross-section graphs** displays separate lines for each cross-section in a single graph. We caution you that both types of panel graphs may become difficult to read when there are large numbers of cross-sections. For example, the individual graphs for the 10 cross-section panel data depicted here provides information on general trends, but little in the way of detail:

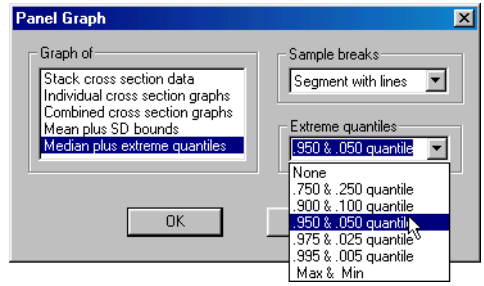

Series: F Workfile: GRUNFELD_BALTAGI_PANEL\Untitled
View Proc Object Properties Print Name Freeze Sample Genr Sheet Stats Ident
F

The remaining two options allow you to plot a single graph containing summary statistics for each period.

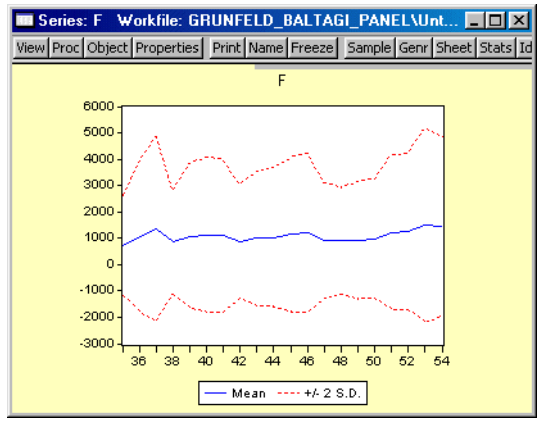
For line graphs, you may select **Mean plus SD bounds**, and then use the drop down menu on the lower right to choose between displaying no bounds, and 1, 2, or 3 standard deviation bounds. For other graph types such as area or spike, you may only display the means of the data by period.



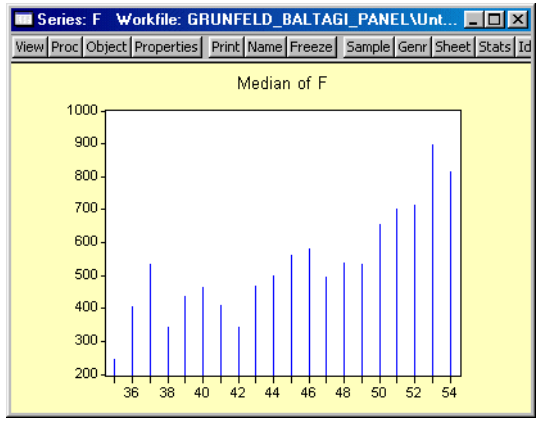
For line graphs you may select Median plus extreme quantiles, and then use the drop down menu to choose additional extreme quantiles to be displayed. For other graph types, only the median may be plotted.



Suppose, for example, that we display a line graph containing the mean and 2 standard deviation bounds for the F series. EViews computes, for each period, the mean and standard deviation of F across cross-sections, and displays these in a time series graph:



Similarly, we may display a spike graph of the medians of F for each period:

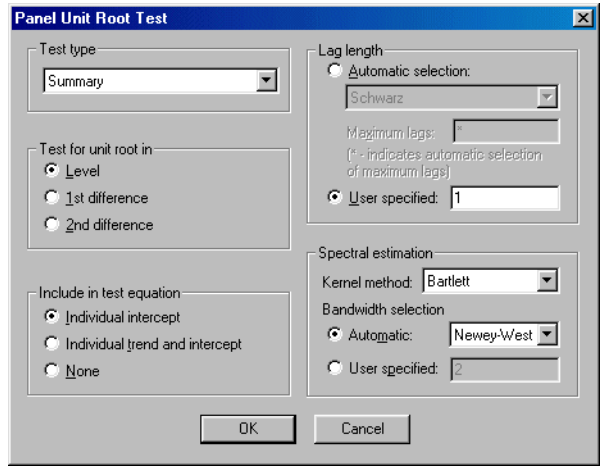


Displaying graph views of a group object in a panel workfile involves similar choices about the handling of the panel structure.

Panel Unit Root Tests

EViews provides convenient tools for computing panel unit root tests. You may compute one or more of the following tests: Levin, Lin and Chu (2002), Breitung (2002), Im, Pesaran and Shin (2003), Fisher-type tests using ADF and PP tests—Maddala and Wu (1999) and Choi (2001), and Hadri (1999).

These tests are described in detail in "Panel Unit Root Tests" beginning on page 530.



To compute the unit root test on a series, simply select **View/Unit Root Test...**from the menu of a series object.

By default, EViews will compute a **Summary** of all of the unit root tests, but you may use the combo box in the upper left hand corner to select an individual test statistic.

In addition, you may use the dialog to specify trend and intercept settings, to specify lag length selection, and to provide details on the spectral estimation used in computing the test statistic or statistics.

To begin, we open the F series in our example panel workfile, and accept the defaults to compute the summary of all of the unit root tests on the level of F. The results are given by

Panel unit root test: Summary
Date: 02/01/04 Time: 10:40
Sample: 1935 1954
Exogenous variables: Individual effects
User specified lags at: 1
Newey-West bandwidth selection using Bartlett kernel
Balanced observations for each test

Method	Statistic	Prob.**	Cross-sections	Obs
Null: Unit root (assumes common unit root process)				
Levin, Lin & Chu t*	1.71727	0.9570	10	180
Breitung t-stat	-3.21275	0.0007	10	170
Null: Unit root (assumes individual unit root process)				
Im, Pesaran and Shin W-stat	-0.51923	0.3018	10	180
ADF - Fisher Chi-square	33.1797	0.0322	10	180
PP - Fisher Chi-square	41.9742	0.0028	10	190
Null: No unit root (assumes common unit root process)				
Hadri Z-stat	3.04930	0.0011	10	200

** Probabilities for Fisher tests are computed using an asympotic Chi -square distribution. All other tests assume asymptotic normality.

Note that there is a fair amount of disagreement in these results as to whether F has a unit root, even within tests that evaluate the same null hypothesis (e.g., Im, Pesaran and Shin vs. the Fisher ADF and PP tests).

To obtain additional information about intermediate results, we may rerun the panel unit root procedure, this time choosing a specific test statistic. Computing the results for the IPS test, for example, displays (in addition to the previous IPS results) ADF test statistic results for each cross-section in the panel:

Intermediate ADF test results

Cross section	t-Stat	Prob.	E(t)	E(Var)	Lag	Max Lag	Obs
1	-2.3596	0.1659	-1.511	0.953	1	1	18
2	-3.6967	0.0138	-1.511	0.953	1	1	18
3	-2.1030	0.2456	-1.511	0.953	1	1	18
4	-3.3293	0.0287	-1.511	0.953	1	1	18
5	0.0597	0.9527	-1.511	0.953	1	1	18
6	1.8743	0.9994	-1.511	0.953	1	1	18
7	-1.8108	0.3636	-1.511	0.953	1	1	18
8	-0.5541	0.8581	-1.511	0.953	1	1	18
9	-1.3223	0.5956	-1.511	0.953	1	1	18
10	-3.4695	0.0218	-1.511	0.953	1	1	18
Average	-1.6711		-1.511	0.953			

Estimation

EViews provides sophisticated tools for estimating equations in your panel structured workfile. See Chapter 29, "Panel Estimation", beginning on page 901 for documentation.

Stacked By-Group Analysis

There are various by-group analysis tools that may be used to perform analysis of panel data. Previously, we considered an example of using by-group tools to examine data in "Cross-section and Period Summaries" on page 887. Standard by-group views may also be to test for equality of means, medians, or variances between groups, or to examine box-plots by cross-section or period.

For example, to compute a test of equality of means for F between firms, simply open the series, then select **View/Test for Descriptive Statistics/Equality Tests by Classification...**. Enter FN in the **Series/Group for Classify** edit field, and select **OK** to continue. EViews will compute and display the results for an ANOVA for F, classifying the data by firm ID. The top portion of the ANOVA results is given by:

Test for Equality of Means of F
Categorized by values of FN
Date: 02/07/04 Time: 23:33
Sample: 1935 1954
Included observations: 200

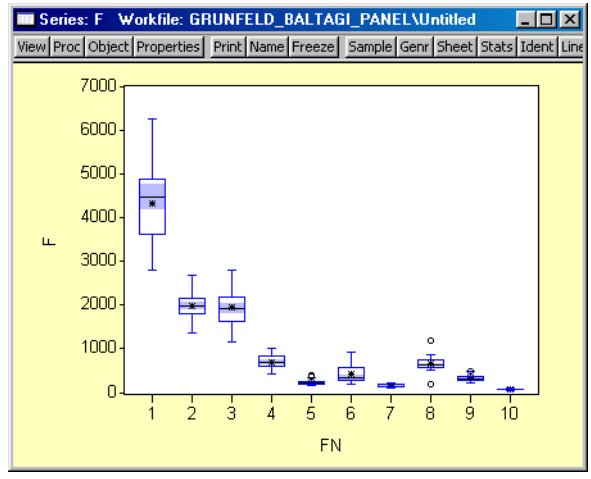
Method	df	Value	Probability
Anova F-statistic	(9, 190)	293.4251	0.0000

Analysis of Variance

Source of Variation	df	Sum of Sq.	Mean Sq.
Between	9	3.21E+08	35640052
Within	190	23077815	121462.2
Total	199	3.44E+08	1727831.

Note in this example that we have relatively few cross-sections with moderate numbers of observations in each firm. Data with very large numbers of group identifiers and few observations are not recommended for this type of testing. To test equality of means between periods, call up the dialog and enter either YEAR or DATEID as the series by which you will classify.

A graphical summary of the primary information in the ANOVA may be obtained by displaying boxplots by cross-section or period. For moderate numbers of distinct classifier values, the graphical display may prove informative. **Select View/ Descriptive Statistics/Boxplot by Classification...** Enter FN in the **Series/Group for Classify** edit field, and select **OK** to display the boxplots using the default settings.

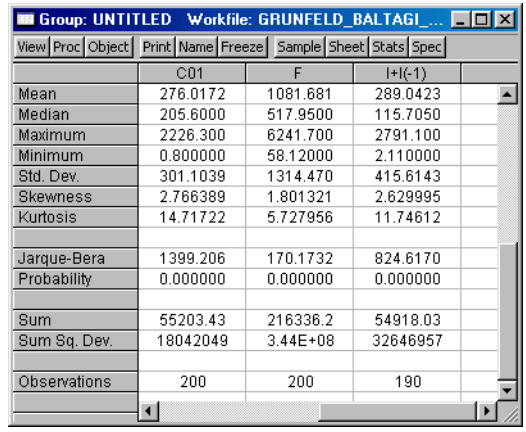


Stacked Analysis

A wide range of analyses are available in panel structured workfiles that have not been specifically redesigned to use the panel structure of your data. These tools allow you to

work with and analyze the stacked data, while taking advantage of the support for handling lags and leads in the panel structured workfile.

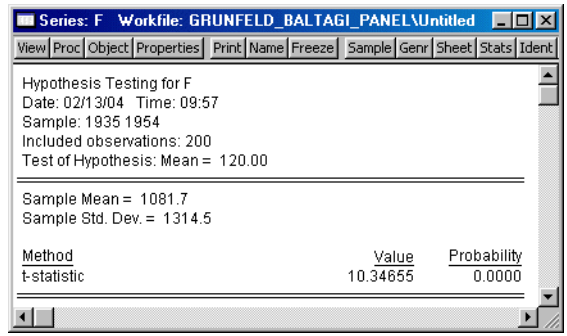
We may, for example, take our example panel workfile, create a group containing the series C01, F, and the expression I + I(-1), and then select **View/Descriptive Stats/Individual Samples** from the group menu. EViews displays the descriptive statistics for the stacked data.



	C01	F	I+I(-1)
Mean	276.0172	1081.681	289.0423
Median	205.6000	517.9500	115.7050
Maximum	2226.300	6241.700	2791.100
Minimum	0.800000	58.12000	2.110000
Std. Dev.	301.1039	1314.470	415.6143
Skewness	2.766389	1.801321	2.629995
Kurtosis	14.71722	5.727956	11.74612
Jarque-Bera	1399.206	170.1732	824.6170
Probability	0.000000	0.000000	0.000000
Sum	55203.43	216336.2	54918.03
Sum Sq. Dev.	18042049	3.44E+08	32646957
Observations	200	200	190

Note that the calculations are performed over the entire 200 observation stacked data, and that the statistics for I + I(-1) use only 190 observations (200 minus 10 observations corresponding to the lag of the first observation for each firm).

Similarly, suppose you wish to perform a hypothesis testing on a single series. Open the window for the series F, and select **View/Tests for Descriptive Stats/Simple Hypothesis Tests...**. Enter "120" in the edit box for testing the mean value of the stacked series against a null of 120. EViews displays the results of a simple hypothesis test for the mean of the 200 observation stacked data.


Series: F Workfile: GRUNFELD_BALTAGI_PANEL\Untitled
View Proc Object Properties Print Name Freeze Sample Genr Sheet Stats Ident
Hypothesis Testing for F
Date: 02/13/04 Time: 09:57
Sample: 1935 1954
Included observations: 200
Test of Hypothesis: Mean = 120.00
Sample Mean = 1081.7
Sample Std. Dev. = 1314.5
Method Value Probability
t-statistic 10.34655 0.0000

While a wide variety of stacked analyses are supported, various views and procedures are not available in panel structured workfiles. You may not, for example, perform seasonal adjustment or estimate VAR or VEC models with the stacked panel.

Chapter 29. Panel Estimation

EViews allows you to estimate panel equations using linear or nonlinear squares or instrumental variables (two-stage least squares), with correction for fixed or random effects in both the cross-section and period dimensions, AR errors, GLS weighting, and robust standard errors. In addition, GMM tools may be used to estimate most of the these specifications with various system-weighting matrices. Specialized forms of GMM also allow you to estimate dynamic panel data specifications. Note that all of the estimators described in this chapter require a panel structured workfile ("Structuring a Panel Workfile" on page 873).

We begin our discussion by briefly outlining the dialog settings associated with common panel equation specifications. While the wide range of models that EViews supports means that we cannot exhaustively describe all of the settings and specifications, we hope to provide you a roadmap of the steps you must take to estimate your panel equation.

More useful, perhaps, is the discussion that follows, which follows the estimation of some simple panel examples, and describes the use of the wizard for specifying dynamic panel data models.

A background discussion of the supported techniques is provided in "Estimation Background" in "Pooled Estimation" on page 859, and in "Estimation Background" beginning on page 930.

Estimating a Panel Equation

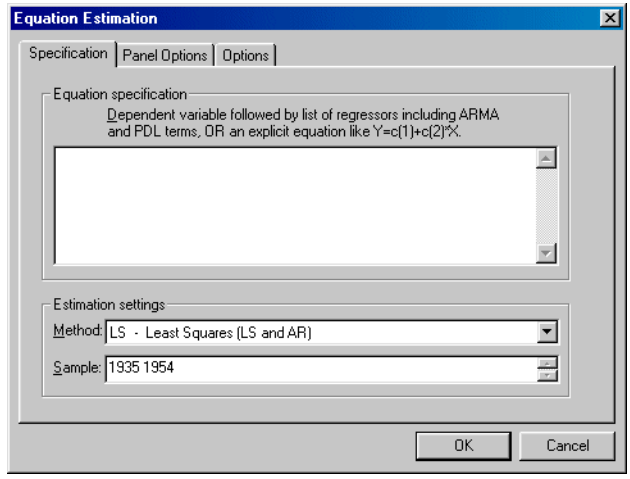
The first step in estimating a panel equation is to call up an equation dialog by clicking on **Object/New Object.../Equation** or **Quick/Estimate Equation...** from the main menu, or typing the keyword "EQUATION" in the command window. You should make certain that your workfile is structured as a panel workfile. EViews will detect the presence of your panel structure and in place of the standard equation dialog will open the panel **Equation Estimation** dialog.

You should use the **Method** combo box to choose between **LS - Least Squares (and AR),** ordinary least squares regression, **TSLS - Two-Stage Least Squares (and AR)**, two-stage least squares (instrumental variable) regression, and **GMM / DPD - Generalized Method of Moments / Dynamic Panel Data** techniques. If you select the either of the latter two methods, the dialog will be updated to provide you with an additional page for specifying instruments (see "Instrumental Variables Estimation" on page 904).

Least Squares Estimation

The basic least squares estimation dialog is a multi-page dialog with pages for the basic specification, panel estimation options, and general estimation options.

Least Squares Specification

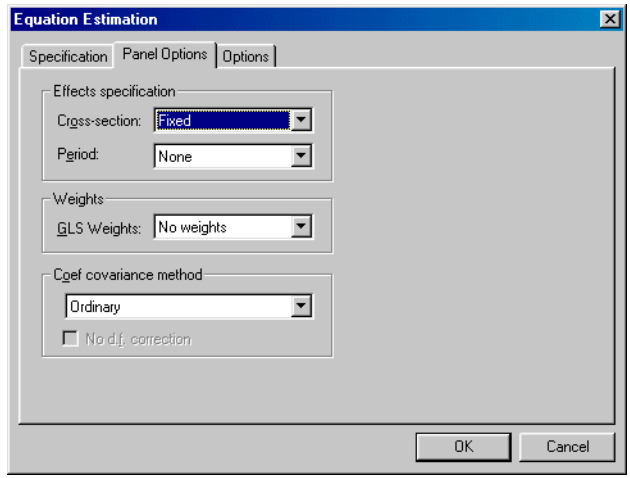


You should provide an equation specification in the upper **Equation specification** edit box, and an estimation sample in the **Sample** edit box.

The equation may be specified by list or by expression as described in "Specifying an Equation in EViews" on page 444.

In general, most of the specifications allowed in non-panel equation settings may also be specified here. You may, for example, include AR terms in both linear and nonlinear specifications, and may include PDL terms in equations specified by list. You may not, however, include MA terms in a panel setting.

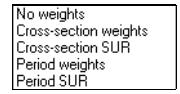
Least Squares Panel Options



Next, click on the **Panel Options** tab to specify additional panel specific estimation settings.

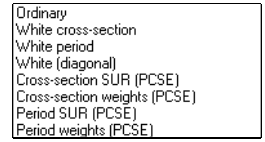
First, you should account for individual and period effects using the **Fixed and Random Effects** combo boxes. By default, EViews assumes that there are no effects so that both combo boxes are set to **None**. You may change the default settings to allow for either **Fixed** or **Random** effects in either the cross-section or period dimension, or both. See the pool discussion of "Fixed and Random Effects" on page 862 for details.

You should be aware that when you select a fixed or random effects specification, EViews will automatically add a constant to the common coefficients portion of the specification if necessary, to ensure that the effects sum to zero.



Next, you should specify settings for **GLS Weights**. You may choose to estimate with no weighting, or with **Cross-section weights**, **Cross-section SUR**, **Period weights**, **Period SUR**. The **Cross-section SUR** setting allows for contemporaneous correlation between cross-sections, while the **Period SUR** allows for general correlation of residuals across periods for a specific cross-section. **Cross-section weights** and **Period weights** allow for heteroskedasticity in the relevant dimension.

For example, if you select **Cross section weights**, EViews will estimate a feasible GLS specification assuming the presence of cross-section heteroskedasticity. If you select **Cross-section SUR**, EViews estimates a feasible GLS specification correcting for both cross-section heteroskedasticity and contemporaneous correlation. Similarly, **Period weights** allows for period heteroskedasticity, while **Period SUR** corrects for both period heteroskedasticity and general correlation of observations within a given cross-section. Note that the SUR specifications are both examples of what is sometimes referred to as the Parks estimator. See the pool discussion of "Generalized Least Squares" on page 864 for additional details.



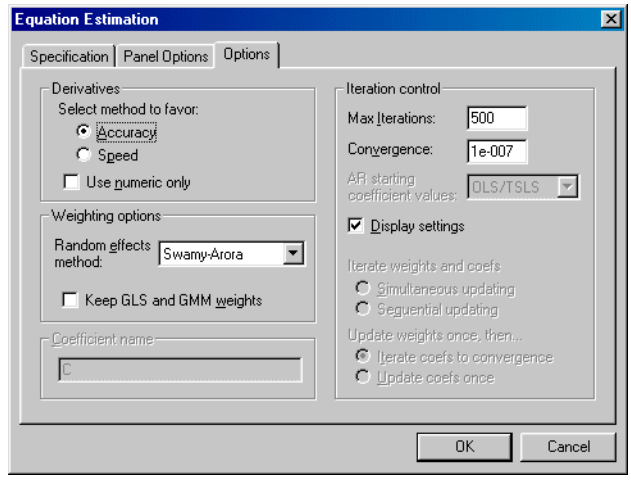
Lastly, you should specify a method for computing coefficient covariances. You may use the combo box labeled **Coef covariance method** to select from the various robust methods available for computing the coefficient standard errors. The covariance calculations may be chosen to be robust under various assumptions, for example, general correlation of observations within a cross-section, or perhaps cross-section heteroskedasticity. Click on the checkbox **No d.f. correction** to perform the calculations without the leading degree of freedom correction term.

Each of the methods is described in greater detail in "Robust Coefficient Covariances" on page 869 of the pool chapter.

You should note that some combinations of specifications and estimation settings are not currently supported. You may not, for example, estimate random effects models with cross-section specific coefficients, AR terms, or weighting. Furthermore, while two-way random effects specifications are supported for balanced data, they may not be estimated in unbalanced designs.

LS Options

Lastly, clicking on the **Options** tab in the dialog brings up a page displaying computational options for panel estimation. Settings that are not currently applicable will be grayed out.

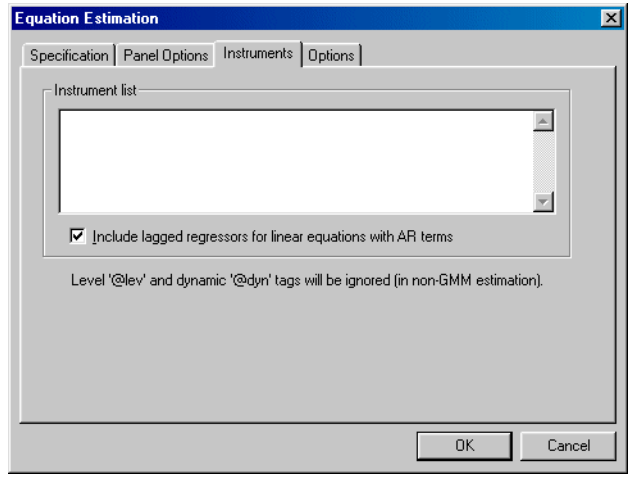


These options control settings for derivative taking, random effects component variance calculation, coefficient usage, iteration control, and the saving of estimation weights with the equation object.

These options are identical to those found in pool equation estimation, and are described in considerable detail in "Options" on page 849.

Instrumental Variables Estimation

To estimate a pool specification using instrumental variables techniques, you should select **TSLS - Two-Stage Least Squares (and AR)** in the **Method** combo box at the bottom of the main (**Specification**) dialog page. EViews will respond by creating a four page dialog in which the third page is used to specify your instruments.



While the three original pages are unaffected by this choice of estimation method, note the presence of the new third dialog page labeled **Instruments**, which you will use to specify your instruments. Click on the **Instruments** tab to display the new page.

IV Instrument Specification

There are only two parts to the instrumental variables

page. First, in the edit box labeled **Instrument list**, you will list the names of the series or groups of series you wish to use as instruments.

Next, if your specification contains AR terms, you should use the checkbox to indicate whether EViews should automatically create instruments to be used in estimation from lags of the dependent and regressor variables in the original specification. When estimating an equation specified by list that contains AR terms, EViews transforms the linear model and estimates the nonlinear differenced specification. By default, EViews will add lagged values of the dependent and independent regressors to the corresponding lists of instrumental variables to account for the modified specification, but if you wish, you may uncheck this option.

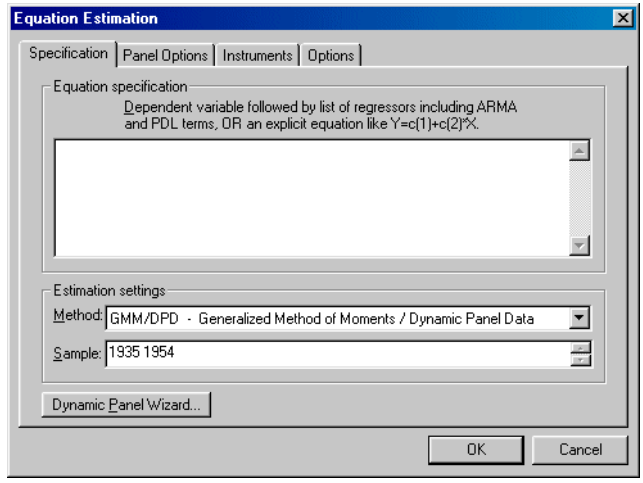
See the pool chapter discussion "Instrumental Variables" on page 867 for additional detail.

GMM Estimation

To estimate a panel specification using GMM techniques, you should select **GMM / DPD - Generalized Method of Moments / Dynamic Panel Data** in the **Method** combo box at the bottom of the main (**Specification**) dialog page. Again, you should make certain that your workfile has a panel structure. EViews will respond by displaying a four page dialog that differs significantly from the previous dialogs.

GMM Specification

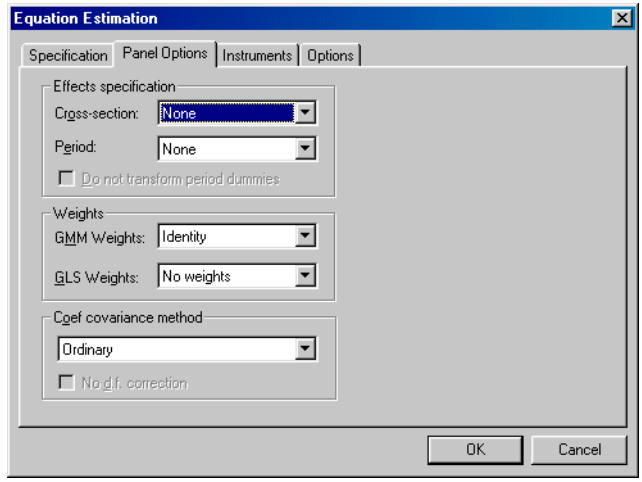
The specification page is similar to the earlier dialogs. Like the earlier dialogs, you will enter your equation specification in the upper edit box and your sample in the lower edit box.



Note, however, the presence of the **Dynamic Panel Wizard...** button on the bottom of the dialog. Pressing this button opens a wizard that will aid you in filling out the dialog so that you may employ dynamic panel data techniques such as the Arellano-Bond 1-step estimator for models with lagged endogenous variables and cross-section fixed effects. We will return to this wizard shortly ("GMM Example" on page 917).

GMM Panel Options

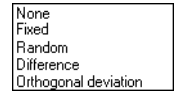
Next, click on the **Panel Options** dialog to specify additional settings for your estimation procedure.



As before, the dialog allows you to indicate the presence of cross-section or period fixed and random effects, to specify GLS weighting, and coefficient covariance calculation methods.

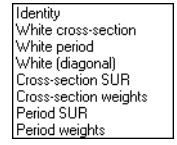
There are, however, notable changes in the available settings.

First, when estimating with GMM, there are two additional choices for handling cross-section fixed effects. These choices allow you to indicate a transformation method for eliminating the effect from the specification.

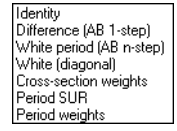


You may select **Difference** to indicate that the estimation procedure should use first differenced data (as in Arellano and Bond, 1991), and you may use **Orthogonal Deviations** (Arellano and Bover, 1995) to perform an alternative method of removing the individual effects.

Second, the dialog presents you with a new combo box so that you may specify weighting matrices that may provide for additional efficiency of GMM estimation under appropriate assumptions. Here, the available options depend on other settings in the dialog.



In most cases, you may select a method that computes weights under one of the assumptions associated with the robust covariance calculation methods (see "Least Squares Panel Options" on page 902). If you select **White cross-section**, for example, EViews uses GMM weights that are formed assuming that there is contemporaneous correlation between cross-sections.



If, however, you account for cross-section fixed effects by performing first difference estimation, EViews provides you with a modified set of GMM weights choices. In particular, the **Difference (AB 1-step)** weights are those associated with the difference transformation. Selecting these weights allows you to estimate the GMM specification typi-

cally referred to as Arellano-Bond 1-step estimation. Similarly, you may choose the **White period (AB 1-step)** weights if you wish to compute Arellano-Bond 2-step or multi-step estimation. Note that the White period weights have been relabeled to indicate that they are typically associated with a specific estimation technique.

Note also that if you estimate your model using difference or orthogonal deviation methods, some GMM weighting methods will no longer be available.

GMM Instruments

Instrument specification in GMM estimation follows the discussion above with a few additional complications.

First, you may enter your instrumental variables as usual by providing the names of series or groups in the edit field. In addition, you may tag instruments as period-specific predetermined instruments, using the "@DYN" keyword, to indicate that the number of implied instruments expands dynamically over time as additional predetermined variables become available.

To specify a set of dynamic instruments associated with the series X, simply enter "@DYN(X)" as an instrument in the list. EViews will, by default, use the series X(-2), X(-3), ..., X(-T), as instruments for each period (where available). Note that the default set of instruments grows very quickly as the number of periods increases. With 20 periods, for example, there are 171 implicit instruments associated with a single dynamic instrument. To limit the number of implied instruments, you may use only a subset of the instruments by specifying additional arguments to "@DYN" describing a range of lags to be used.

For example, you may limit the maximum number of lags to be used by specifying both a minimum and maximum number of lags as additional arguments. The instrument specification:

```
@dyn(x, -2, -5)
```

instructs EViews to include lags of X from 2 to 5 as instruments for each period.

If a single argument is provided, EViews will use it as the minimum number of lags to be considered, and will include all higher ordered lags. For example:

```
@dyn(x, -5)
```

includes available lags of X from 5 to the number of periods in the sample.

Second, in specifications estimated using transformations to remove the cross-section fixed effects (first differences or orthogonal deviations), use may use the "@LEV" keyword to instruct EViews to use the instrument in untransformed, or level form. Tagging an instrument with "@LEV" indicates that the instrument is for the transformed equation If

"@LEV" is not provided, EViews will transform the instrument to match the equation transformation.

If, for example, you estimate an equation that uses orthogonal deviations to remove a cross-section fixed effect, EViews will, by default, compute orthogonal deviations of the instruments provided prior to their use. Thus, the instrument list:

```
z1 z2 @lev(z3)
```

will use the transformed Z1 and Z2, and the original Z3 as the instruments for the specification.

Note that in specifications where "@DYN" and "@LEV" keywords are not relevant, they will be ignored. If, for example, you first estimate a GMM specification using first differences with both dynamic and level instruments, and then re-estimate the equation using LS, EViews will ignore the keywords, and use the instruments in their original forms.

GMM Options

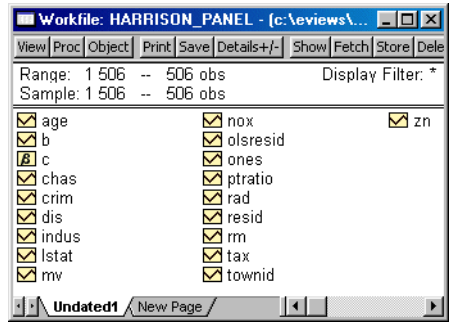
Lastly, clicking on the **Options** tab in the dialog brings up a page displaying computational options for GMM estimation. These options are virtually identical to those for both LS and IV estimation (see "LS Options" on page 904). The one difference is in the option for saving estimation weights with the object. In the GMM context, this option applies to both the saving of GLS as well as GMM weights.

Panel Estimation Examples

Least Squares Examples

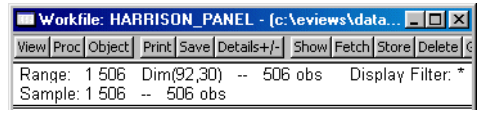
To illustrate the estimation of panel equations in EViews, we first consider an example involving unbalanced panel data from Harrison and Rubinfeld (1978) for the study of hedonic pricing. The data are well known and used as an example dataset in many sources (*e.g.*, Baltagi (2001), p. 166).

The data consist of 506 census tract observations on 92 towns in the Boston area with group sizes ranging from 1 to 30. The dependent variable of interest is the median value of owner occupied towns (MV), and the regressors include various measures of housing desirability.



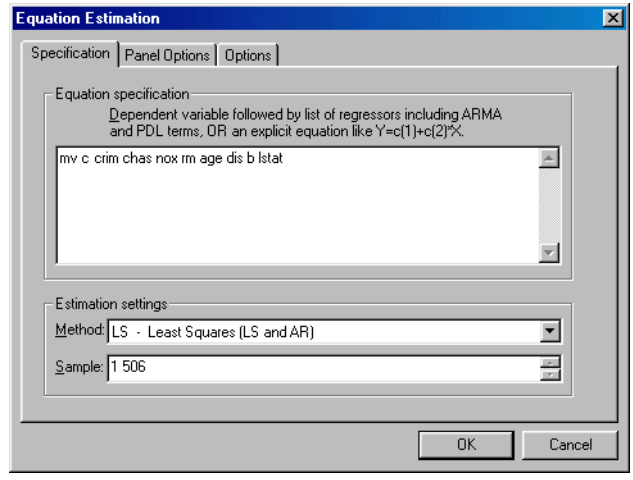
We begin our example by structuring our workfile as an undated panel. Click on the "Range:" description in the workfile window,

select **Undated Panel**, and enter "TOWNID" as the **Identifier series**. EViews will prompt you twice to create a CELLID series to uniquely identify observations. Click on **OK** to both questions to accept your settings.


Workfile: HARRISON_PANEL - (c:\eviews\data...
View Proc Object Print Save Details+/- Show Fetch Store Delete
Range: 1 506 Dim(92,30) -- 506 obs Display Filter: *
Sample: 1 506 -- 506 obs

EViews restructures your workfile so that it is an unbalanced panel workfile. The top portion of the workfile window will change to show the undated structure which has 92 cross-sections and a maximum of 30 observations in a cross-section.

Next, we open the equation specification dialog by selecting **Quick/Estimate Equation** from the main EViews menu.


Equation Estimation
Specification
Panel Options
Options
Equation specification
Dependent variable followed by list of regressors including ARMA and PDL terms, OR an explicit equation like Y=c(1)+c(2)*X.
mv c crim chas nox rm age dis b lstat
Estimation settings
Method: LS - Least Squares (LS and AR)
Sample: 1 506
OK
Cancel

First, following Baltagi and Chang (1994) (also described in Baltagi, 2001), we estimate a fixed effects specification of a hedonic housing equation. The dependent variable in our specification is the median value MV, and the regressors are the crime rate (CRIM), a dummy variable for the property along Charles River (CHAS), air pollution (NOX), average number of rooms (RM), proportion of older units (AGE), distance from employment centers (DIS), proportion of African-Americans in the population (B), and the proportion of lower status individuals (LSTAT). Note that you may include a constant term C in the specification. Since we are estimating a fixed effects specification, EViews will add one if it is not present so that the fixed effects estimates are relative to the constant term and add up to zero.

Click on the **Panel Options** tab and select **Fixed** for the **Cross-section** effects. To match the Baltagi and Chang results, we will leave the remaining settings at their defaults. Click on **OK** to accept the specification.

Dependent Variable: MV
Method: Panel Least Squares
Date: 02/16/04 Time: 12:07
Sample: 1 506
Cross-sections included: 92
Total panel (unbalanced) observations: 506

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	8.993272	0.134738	66.74632	0.0000
CRIM	-0.625400	0.104012	-6.012746	0.0000
CHAS	-0.452414	0.298531	-1.515467	0.1304
NOX	-0.558938	0.135011	-4.139949	0.0000
RM	0.927201	0.122470	7.570833	0.0000
AGE	-1.406955	0.486034	-2.894767	0.0040
DIS	0.801437	0.711727	1.126045	0.2608
B	0.663405	0.103222	6.426958	0.0000
LSTAT	-2.453027	0.255633	-9.595892	0.0000

Effects Specification

Cross-section fixed (dummy variables)

R-squared	0.918370	Mean dependent var	9.942268
Adjusted R-squared	0.898465	S.D. dependent var	0.408758
S.E. of regression	0.130249	Akaike info criterion	-1.063668
Sum squared resid	6.887683	Schwarz criterion	-0.228384
Log likelihood	369.1080	F-statistic	46.13805
Durbin-Watson stat	1.999986	Prob(F-statistic)	0.000000

The results for the fixed effects estimation are depicted here. Note that as in pooled estimation, the reported R-squared and F-statistics are based on the difference between the residuals sums of squares from the estimated model, and the sums of squares from a *single* constant-only specification, not from a fixed-effect-only specification. Similarly, the reported information criteria report likelihoods adjusted for the number of estimated coefficients, including fixed effects. Lastly, the reported Durbin-Watson stat is formed simply by computing the first-order residual correlation on the stacked set of residuals.

We may click on the **Estimate** button to modify the specification to match the Wallace-Hussain random effects specification considered by Baltagi and Chang. We modify the specification to include the additional regressors used in estimation, change the cross-section effects to be estimated as a random effect, and use the **Options** page to set the random effects computation method to Wallace-Hussain.

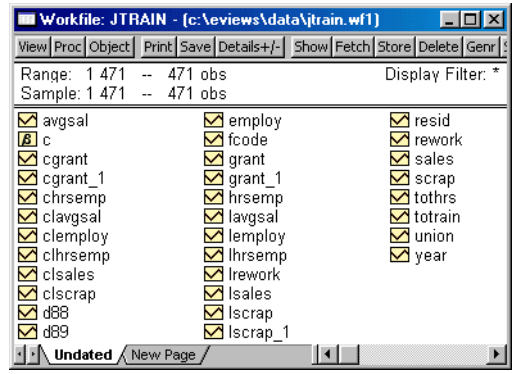
The top portion of the resulting output is given by:

Dependent Variable: MV
Method: Panel EGLS (Cross-section random effects)
Date: 02/16/04 Time: 12:28
Sample: 1 506
Cross-sections included: 92
Total panel (unbalanced) observations: 506
Wallace and Hussain estimator of component variances

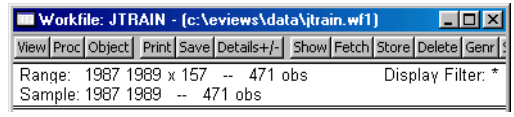
Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	9.684427	0.207691	46.62904	0.0000
CRIM	-0.737616	0.108966	-6.769233	0.0000
ZN	0.072190	0.684633	0.105443	0.9161
INDUS	0.164948	0.426376	0.386860	0.6990
CHAS	-0.056459	0.304025	-0.185703	0.8528
NOX	-0.584667	0.129825	-4.503496	0.0000
RM	0.908064	0.123724	7.339410	0.0000
AGE	-0.871415	0.487161	-1.788760	0.0743
DIS	-1.423611	0.462761	-3.076343	0.0022
RAD	0.961362	0.280649	3.425493	0.0007
TAX	-0.376874	0.186695	-2.018658	0.0441
PTRATIO	-2.951420	0.958355	-3.079674	0.0022
B	0.565195	0.106121	5.325958	0.0000
LSTAT	-2.899084	0.249300	-11.62891	0.0000
Effects Specification				
Cross-section random S.D. / Rho			0.126983	0.4496
Idiosyncratic random S.D. / Rho			0.140499	0.5504

Note that the estimates of the component standard deviations must be squared to match the component variances reported by Baltagi and Chang (0.016 and 0.020, respectively).

Next, we consider an example of estimation with standard errors that are robust to serial correlation. For this example, we employ data on job training grants used in examples from Wooldridge (2002, p. 276 and 282).



As before, the first step is to structure the workfile as a panel workfile. Click on **Range:** to bring up the dialog, and enter "YEAR" as the date identifier and "FCODE" as the cross-section ID.

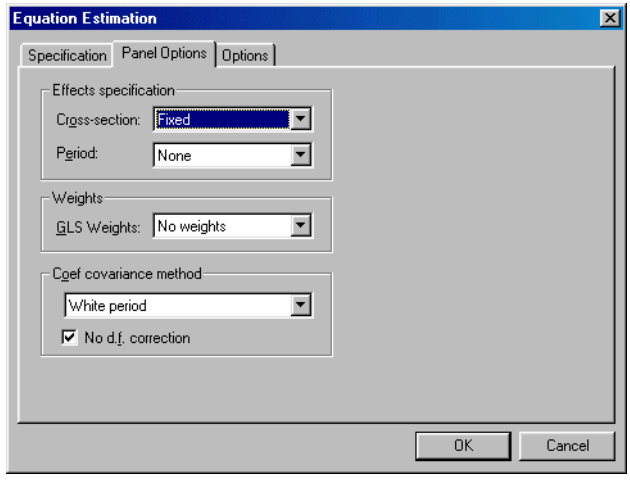

Workfile: JTRAIN - (c:\eviews\data\jtrain.wf1)
View Proc Object Print Save Details+/- Show Fetch Store Delete Genr
Range: 1987 1989 x 157 -- 471 obs
Display Filter: *
Sample: 1987 1989 -- 471 obs

EViews will structure the workfile so that it is a panel workfile with 157 cross-sections, and three annual observations. Note that even though there are 471 observations in the workfile, a large number of them contain missing values for variables of interest.

To estimate the fixed effect specification with robust standard errors (Wooldridge example 10.5, p. 276), click on specification **Quick/Estimate Equation** from the main EViews menu. Enter the list specification:

```
lscrap c d88 d89 grant grant_1
```

in the **Equation specification** edit box on the main page and select **Fixed** in the Cross-section effects specification combo box on the **Panel Options** page. Lastly, since we wish to compute standard errors that are robust to serial correlation (Arellano (1987), White (1984)), we choose **White period** as the **Coef covariance method**. To match the reported Wooldridge example, we must select **No d.f. correction** in the covariance calculation.



Click on **OK** to accept the options. EViews displays the results from estimation:

Dependent Variable: LSCRAP
Method: Panel Least Squares
Date: 02/16/04 Time: 13:28
Sample: 1987 1989
Cross-sections included: 54
Total panel (balanced) observations: 162
White period standard errors & covariance (no d.f. correction)

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.597434	0.062489	9.560565	0.0000
D88	-0.080216	0.095719	-0.838033	0.4039
D89	-0.247203	0.192514	-1.284075	0.2020
GRANT	-0.252315	0.140329	-1.798022	0.0751
GRANT_1	-0.421589	0.276335	-1.525648	0.1301

Effects Specification

Cross-section fixed (dummy variables)

R-squared	0.927572	Mean dependent var	0.393681
Adjusted R-squared	0.887876	S.D. dependent var	1.486471
S.E. of regression	0.497744	Akaike info criterion	1.715383
Sum squared resid	25.76593	Schwarz criterion	2.820819
Log likelihood	-80.94602	F-statistic	23.36680
Durbin-Watson stat	1.996983	Prob(F-statistic)	0.000000

Note that EViews automatically adjusts for the missing values in the data. There are only 162 observations on 54 cross-sections used in estimation. The top portion of the output indicates that the results use robust White period standard errors with no d.f. correction.

Alternately, we may estimate a first difference estimator for these data with robust standard errors (Wooldridge example 10.6, p. 282). Open a new equation dialog by clicking on **Quick/Estimate Equation...**, or modify the existing equation by clicking on the **Estimate** button on the equation toolbar. Enter the specification:

```
d(lscrap) c d89 d(grant) d(grant_1)
```

in the **Equation specification** edit box on the main page, select **None** in the Cross-section effects specification combo box, and **White period** with **No d.f. correction** for the coefficient covariance method on the **Panel Options** page. The reported results are given by:

Dependent Variable: D(LSCRAP)
Method: Panel Least Squares
Date: 02/16/04 Time: 14:51
Sample (adjusted): 1988 1988
Cross-sections included: 54
Total panel (balanced) observations: 108
White period standard errors & covariance (d.f. corrected)

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.090607	0.089760	-1.009442	0.3151
D89	-0.096208	0.113117	-0.850519	0.3970
D(GRANT)	-0.222781	0.131030	-1.700236	0.0921
D(GRANT_1)	-0.351246	0.269704	-1.302339	0.1957

R-squared	0.036518	Mean dependent var	-0.221132
Adjusted R-squared	0.008725	S.D. dependent var	0.579248
S.E. of regression	0.576716	Akaike info criterion	1.773399
Sum squared resid	34.59049	Schwarz criterion	1.872737
Log likelihood	-91.76352	F-statistic	1.313929
Durbin-Watson stat	1.498132	Prob(F-statistic)	0.273884

While current versions of EViews do not provide a full set of specification tests for panel equations, it is a straightforward task to construct some tests using residuals obtained from the panel estimation.

To continue with the Wooldridge example, we may test for AR(1) serial correlation in the first-differenced equation by regressing the residuals from this specification on the lagged residuals using data for the year 1989. First, we save the residual series in the workfile. Click on **Proc/Make Residual Series...** on the estimated equation toolbar, and save the residuals to the series RESID01.

Next, regress RESID01 on RESID01(-1), yielding:

Dependent Variable: RESID01
Method: Panel Least Squares
Date: 02/16/04 Time: 14:57
Sample (adjusted):
Cross-sections included: 54
Total panel (balanced) observations: 54

Variable	Coefficient	Std. Error	t-Statistic	Prob.
RESID01(-1)	0.236906	0.133357	1.776481	0.0814

R-squared	0.056199	Mean dependent var	2.06E-18
Adjusted R-squared	0.056199	S.D. dependent var	0.571061
S.E. of regression	0.554782	Akaike info criterion	1.677863
Sum squared resid	16.31252	Schwarz criterion	1.714696
Log likelihood	-44.30230	Durbin-Watson stat	0.000000

Under the null hypothesis that the original idiosyncratic errors are uncorrelated, the residuals from this equation should have an autocorrelation coefficient of -0.5. Here, we obtain an estimate of $\hat{\rho}_1 = 0.237$ which appears to be far from the null value. A formal Wald hypothesis test rejects the null that the original idiosyncratic errors are serially uncorrelated:

Wald Test:
Equation: Untitled

Test Statistic	Value	df	Probability
F-statistic	26.60396	(1, 53)	0.0000
Chi-square	26.60396	1	0.0000

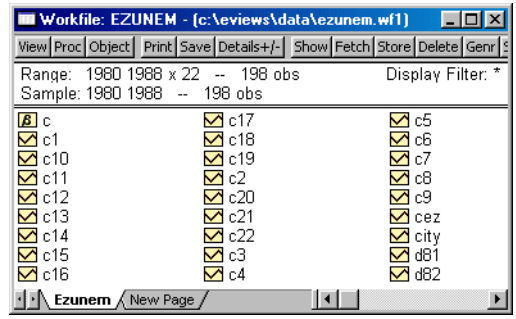
Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
0.5 + C(1)	0.736906	0.142869

Restrictions are linear in coefficients.

Instrumental Variables Example

To illustrate the estimation of instrumental variables panel estimators, we consider an example taken from Papke (1994) for enterprise zone data for 22 communities in Indiana that is outlined in Wooldridge (2002, p. 306).



The panel workfile for this example is structured using YEAR as the period identifier, and CITY as the cross-section identifier. The result is a balanced annual panel for dates from 1980 to 1988 for 22 cross-sections.

To estimate the example specification, create a new equation by typing "TSLS" in the command line, or by clicking on **Quick/Estimate Equation...** in the main menu. Selecting **TSLS - Two-Stage Least Squares (and AR)** in the **Method** combo box to display the instrumental variables estimator dialog, if necessary, and enter:

```
d(luclms) c d(luclms(-1)) d(ez)
```

to regress the difference of log unemployment claims (LUCLMS) on the lag difference, and the difference of enterprise zone designation (EZ). Since the model is estimated with time intercepts, you should click on the **Panel Options** page, and select **Fixed** for the Period effects.

Next, click on the Instruments tab, and add the names:

```
c d(luclms(-2)) d(ez)
```

to the **Instrument list** edit box. Note that adding the constant C to the regressor and instrument boxes is not required since the fixed effects estimator will add it for you. Click on **OK** to accept the dialog settings. EViews displays the output for the IV regression:

Dependent Variable: D(LUCLMS)
Method: Panel Two-Stage Least Squares
Date: 02/16/04 Time: 17:11
Sample (adjusted): 1983 1988
Cross-sections included: 22
Total panel (balanced) observations: 132
Instrument list: C D(LUCLMS(-2)) D(EZ)

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.201654	0.040473	-4.982442	0.0000
D(LUCLMS(-1))	0.164699	0.288444	0.570992	0.5690
D(EZ)	-0.218702	0.106141	-2.060493	0.0414

Effects Specification

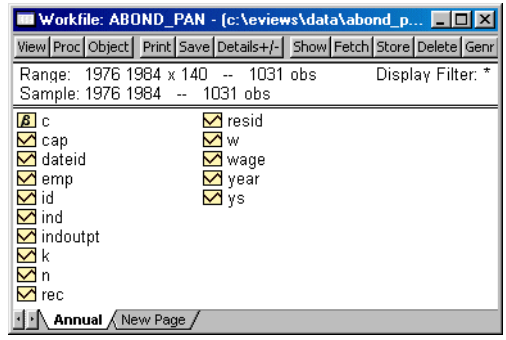
Period fixed (dummy variables)

R-squared	0.280533	Mean dependent var	-0.235098
Adjusted R-squared	0.239918	S.D. dependent var	0.267204
S.E. of regression	0.232956	Sum squared resid	6.729300
Durbin-Watson stat	2.857769	J-statistic	9.39E-29
Instrument rank	8.000000		

Note that the instrument rank in this equation is 8 since the period dummies also serve as instruments, so you have the 3 instruments specified explicitly, plus 5 for the non-collinear period dummy variables.

GMM Example

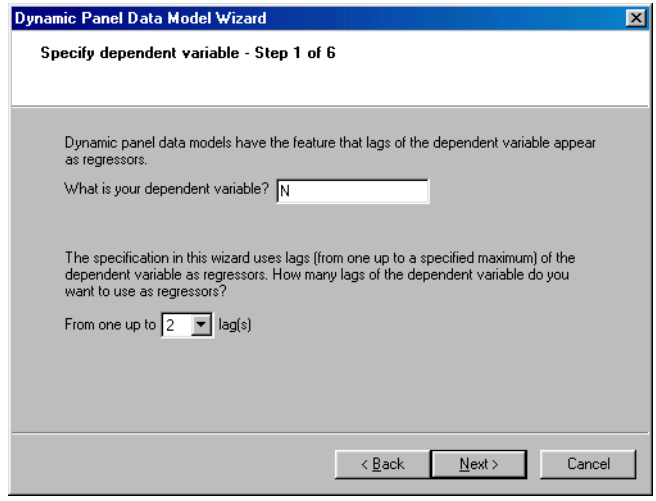
To illustrate the estimation of dynamic panel data models using GMM, we employ the unbalanced 1031 observation panel of firm level data from Layard and Nickell (1986), previously examined by Arellano and Bond (1991). The analysis fits the log of employment (N) to the low of the real wage (W), log of the capital stock (K), and the log of industry output (YS).



The workfile is structured as a dated annual panel using ID as the cross-section identifier series and YEAR as the date classification series.

Since the model is assumed to be dynamic, we employ EViews tools for estimating dynamic panel data models. To bring up the GMM dialog, enter "GMM" in the command line, or select **Quick/Estimate Equation...** from the main menu, and choose **GMM/DPD - Generalized Method of Moments /Dynamic Panel Data** in the **Method** combo box to display the IV estimator dialog.

Click on the button labeled **Dynamic Panel Wizard...** to bring up the DPD wizard. The DPD wizard is a tool that will aid you in filling out the general GMM dialog. The first page is an introductory screen describing the basic purpose of the wizard. Click **Next** to continue.

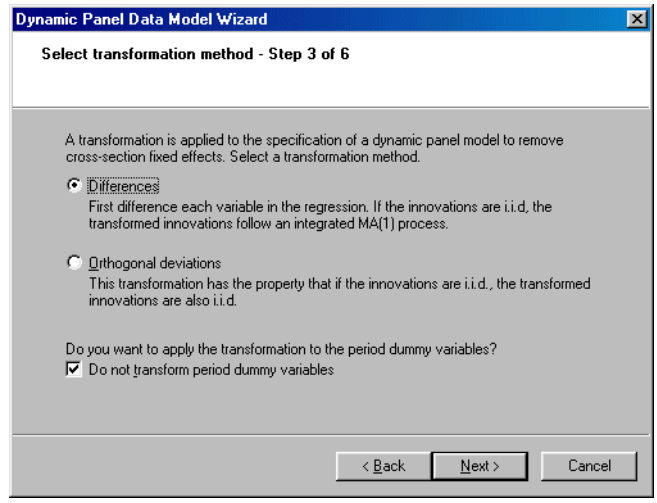

Dynamic Panel Data Model Wizard
Specify dependent variable - Step 1 of 6
Dynamic panel data models have the feature that lags of the dependent variable appear as regressors.
What is your dependent variable? N
The specification in this wizard uses lags (from one up to a specified maximum) of the dependent variable as regressors. How many lags of the dependent variable do you want to use as regressors?
From one up to 2 lag(s)
< Back
Next >
Cancel

The second page of the wizard prompts you for the dependent variable and the number of its lags to include as explanatory variables. In this example, we wish to estimate an equation with N as the dependent variable and N(-1) and N(-2) as explanatory variables so we enter "N" and select "2" lags in the combo box. Click on **Next** to continue to the next page, where you will specify the remaining explanatory variables.

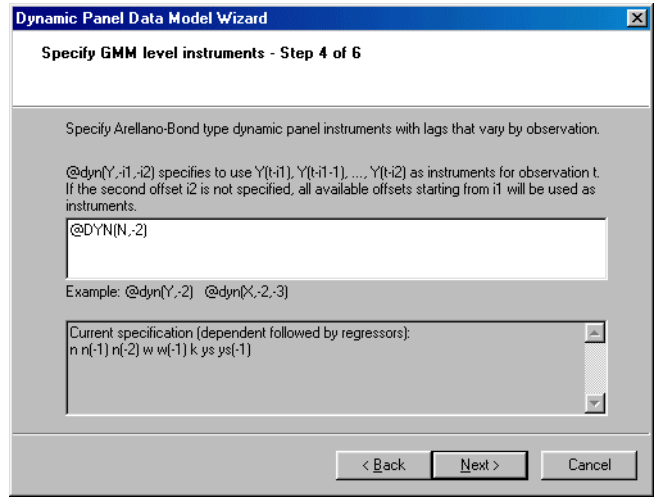
In the third page, you will complete the specification of your explanatory variables. First, enter the list:

```
w w(-1) k ys ys(-1)
```

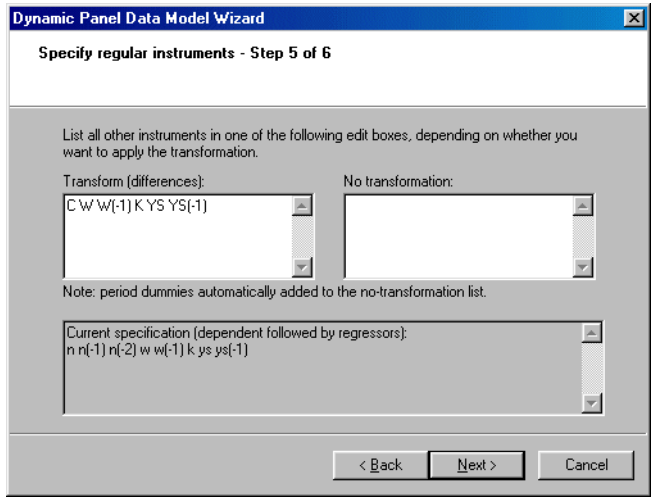
in the regressor edit box to include these variables. Since the desired specification will include time dummies, make certain that the checkbox for **Include period dummy variables** is selected, then click on **Next** to proceed.


Dynamic Panel Data Model Wizard
Select transformation method - Step 3 of 6
A transformation is applied to the specification of a dynamic panel model to remove cross-section fixed effects. Select a transformation method.
Differences
First difference each variable in the regression. If the innovations are i.i.d, the transformed innovations follow an integrated MA(1) process.
Orthogonal deviations
This transformation has the property that if the innovations are i.i.d., the transformed innovations are also i.i.d.
Do you want to apply the transformation to the period dummy variables?
Do not transform period dummy variables
< Back
Next >
Cancel

The next page of the wizard is used to specify a transformation to remove the cross-section fixed effect. You may choose to use first **Differences** or **Orthogonal deviations**. In addition, if your specification includes period dummy variables, there is a checkbox asking whether you wish to transform the period dummies, or to enter them in levels. Here we specify the first difference transformation, and choose to include untransformed period dummies in the transformed equation. Click on **Next** to continue.


Dynamic Panel Data Model Wizard
Specify GMM level instruments - Step 4 of 6
Specify Arellano-Bond type dynamic panel instruments with lags that vary by observation.
@dyn(Y,-i1,-i2) specifies to use Y(t-i1), Y(t-i1-1), ..., Y(t-i2) as instruments for observation t. If the second offset i2 is not specified, all available offsets starting from i1 will be used as instruments.
@DYN(N,-2)
Example: @dyn(Y,-2) @dyn(X,-2,-3)
Current specification (dependent followed by regressors):
n n(-1) n(-2) w w(-1) k ys ys(-1)
< Back
Next >
Cancel

The next page is where you will specify your dynamic period-specific (predetermined) instruments. The instruments should be entered with the "@DYN" tag to indicate that they are to be expanded into sets of predetermined instruments, with optional arguments to indicate the lags to be included. If no arguments are provided, the default is to include all valid lags (from -2 to "-infinity"). Here, we instruct EViews that we wish to use the default lags for N as predetermined instruments.

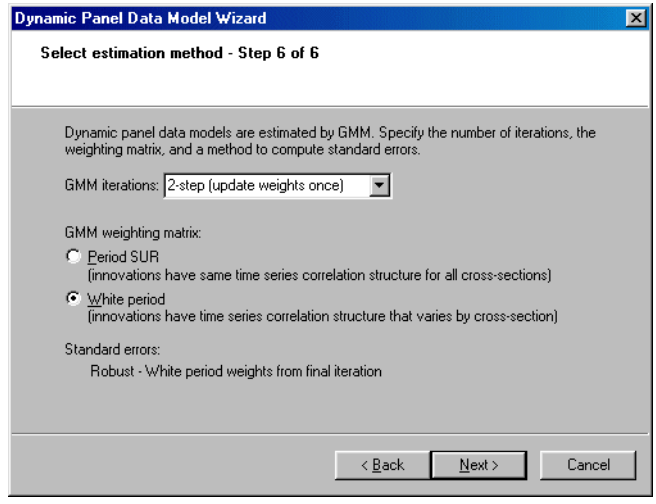

Dynamic Panel Data Model Wizard
Specify regular instruments - Step 5 of 6
List all other instruments in one of the following edit boxes, depending on whether you want to apply the transformation.
Transform (differences):
C W W(-1) K YS YS(-1)
No transformation:
Note: period dummies automatically added to the no-transformation list.
Current specification (dependent followed by regressors):
n n(-1) n(-2) w w(-1) k ys ys(-1)
< Back
Next >
Cancel

You should now specify the remaining instruments. There are two lists that should be provided. The first list, which is entered in the edit field labeled **Transform**, should contain a list of the strictly exogenous instruments that you wish to transform prior to use in estimating the transformed equation. The second list, which should be entered in the **No transform** edit box should contain a list of instruments that should be used directly without transformation. Enter the remaining instruments:

```
w w(-1) k ys ys(-1)
```

in the first edit box and click on **Next** to proceed to the final page.

The final page allows you to specify your GMM weighting and coefficient covariance calculation choices. In the first combo box, you will choose a GMM Iteration option. You may select **1-step (for i.i.d. innovations)** to compute the Arellano-Bond 1-step estimator, **2-step (update weights once),** to compute the Arellano-Bond 2-step estimator, **or n-step (iterate**


Dynamic Panel Data Model Wizard
Select estimation method - Step 6 of 6
Dynamic panel data models are estimated by GMM. Specify the number of iterations, the weighting matrix, and a method to compute standard errors.
GMM iterations:
2-step (update weights once)
GMM weighting matrix:
Period SUR
(innovations have same time series correlation structure for all cross-sections)
White period
(innovations have time series correlation structure that varies by cross-section)
Standard errors:
Robust - White period weights from final iteration
< Back
Next >
Cancel

to convergence), to iterate the weight calculations. In the first case, EViews will provide you with choices for computing the standard errors, but here only White period robust standard errors are allowed. Clicking on **Next** takes you to the final page. Click on **Finish** to return to the **Equation Estimation** dialog.

EViews has filled out the **Equation Estimation** dialog with our choices from the DPD wizard. You should take a moment to examine the settings that have been filled out for you since, in the future, you may wish to enter the specification directly into the dialog without using the wizard. You may also, of course, modify the settings in the dialog prior to continuing. For example, click on the **Panel Options** tab and check the **No d.f. correction** setting in the covariance calculation to match the original Arellano-Bond results (Table 4(b), p. 290). Click on **OK** to estimate the specification.

The top portion of the output describes the estimation settings, coefficient estimates, and summary statistics. Note that both the weighting matrix and covariance calculation method used are described in the top portion of the output.

Dependent Variable: N
Method: Panel Generalized Method of Moments
Transformation: First Differences
Date: 05/29/03 Time: 11:50
Sample (adjusted): 1978 1984
Number of cross-sections used: 140
Total panel (unbalanced) observations: 611
White period instrument weighting matrix
Linear estimation after one-step weighting
Instrument list: W W(-1) K YS YS(-1) @DYN(N) @LEV(@SYSPER)

Variable	Coefficient	Std. Error	t-Statistic	Prob.
N(-1)	0.474150	0.085303	5.558409	0.0000
N(-2)	-0.052968	0.027284	-1.941324	0.0527
W	-0.513205	0.049345	-10.40027	0.0000
W(-1)	0.224640	0.080063	2.805796	0.0052
K	0.292723	0.039463	7.417748	0.0000
YS	0.609775	0.108524	5.618813	0.0000
YS(-1)	-0.446371	0.124815	-3.576272	0.0004
@LEV(@ISPERIOD("1979"))	0.010509	0.007251	1.449224	0.1478
@LEV(@ISPERIOD("1980"))	0.014142	0.009959	1.420077	0.1561
@LEV(@ISPERIOD("1981"))	-0.040453	0.011551	-3.502122	0.0005
@LEV(@ISPERIOD("1982"))	-0.021640	0.011891	-1.819843	0.0693
@LEV(@ISPERIOD("1983"))	-0.001847	0.010412	-0.177358	0.8593
@LEV(@ISPERIOD("1984"))	-0.010221	0.011468	-0.891270	0.3731

The standard errors that we report here are the standard Arellano-Bond 2-step estimator standard errors. Note that there is evidence in the literature that the standard errors for the two-step estimator may not be reliable.

The bottom portion of the output displays additional information about the specification and summary statistics:

Effects Specification			
Cross-section fixed (first differences) Period fixed (dummy variables)			
R-squared	0.384678	Mean dependent var	-0.055606
Adjusted R-squared	0.372331	S.D. dependent var	0.146724
S.E. of regression	0.116243	Sum squared resid	8.080432
Durbin-Watson stat	2.844824	J-statistic	30.11247
Instrument rank	38.00000		

Note in particular the results labeled "J-statistic" and "Instrument rank". Since the reported J-statistic is simply the Sargan statistic (value of the GMM objective function at estimated parameters), and the instrument rank of 38 is greater than the number of estimated coefficients (13), we may use it to construct the Sargan test of over-identifying restrictions. It is worth noting here that the J-statistic reported by a panel equation differs from that reported by an ordinary equation by a factor equal to the number of observations. Under the null hypothesis that the over-identifying restrictions are valid, the Sargan statistic is distributed as a $\chi(k-p)$, where k is the number of estimated coefficients and p is the instrument rank. The p-value of 0.22 in this example may be computed using "scalar pval = @chisq(30.11247, 25)".

Panel Equation Testing

Omitted Variables Test

You may perform an F-est of the joint significance of variables that are presently omitted from a panel or pool equation estimated by list. Select **View/Coefficient Tests/Omitted Variables - Likelihood Ratio...** and in the resulting dialog, enter the names of the variables you wish to add to the default specification. If estimating in a pool setting, you should enter the desired pool or ordinary series in the appropriate edit box (common, cross-section specific, period specific).

When you click on **OK**, EViews will first estimate the unrestricted specification, then form the usual F-est, and will display both the test results as well as the results from the unrestricted specification in the equation or pool window.

Adapting Example 10.6 from Wooldridge (2002, p. 282) slightly, we may first estimate a pooled sample equation for a model of the effect of job training grants on LSCRAP using first differencing. The restricted set of explanatory variables includes a constant and D89. The results from the restricted estimator are given by:

Dependent Variable: D(LSCRAP)
Method: Panel Least Squares
Date: 11/24/04 Time: 09:15
Sample (adjusted): 1988 1989
Cross-sections included: 54
Total panel (balanced) observations: 108

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.168993	0.078872	-2.142622	0.0344
D89	-0.104279	0.111542	-0.934881	0.3520

R-squared	0.008178	Mean dependent var	-0.221132
Adjusted R-squared	-0.001179	S.D. dependent var	0.579248
S.E. of regression	0.579589	Akaike info criterion	1.765351
Sum squared resid	35.60793	Schwarz criterion	1.815020
Log likelihood	-93.32896	F-statistic	0.874003
Durbin-Watson stat	1.445487	Prob(F-statistic)	0.351974

We wish to test the significance of the first differences of the omitted job training grant variables GRANT and GRANT_1. Click on **View/Coefficient Tests/Omitted Variables - Likelihood Ratio...** and type "D(GRANT)" and "D(GRANT_1)" to enter the two variables in differences. Click on **OK** to display the omitted variables test results.

The top portion of the results contains a brief description of the test, the test statistic values, and the associated significance levels:

Omitted Variables: D(GRANT) D(GRANT_1)

F-statistic	1.529525	Prob. F(2,104)	0.221471
Log likelihood ratio	3.130883	Prob. Chi-Square(2)	0.208996

Here, the test statistics do not reject, at conventional significance levels, the null hypothesis that D(GRANT) and D(GRANT_1) are jointly irrelevant.

The bottom portion of the results shows the test equation which estimates under the unrestricted alternative:

Test Equation:
Dependent Variable: D(LSCRAP)
Method: Panel Least Squares
Date: 11/24/04 Time: 09:52
Sample: 1988 1989
Cross-sections included: 54
Total panel (balanced) observations: 108

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.090607	0.090970	-0.996017	0.3216
D89	-0.096208	0.125447	-0.766923	0.4449
D(GRANT)	-0.222781	0.130742	-1.703970	0.0914
D(GRANT_1)	-0.351246	0.235085	-1.494124	0.1382

R-squared	0.036518	Mean dependent var	-0.221132
Adjusted R-squared	0.008725	S.D. dependent var	0.579248
S.E. of regression	0.576716	Akaike info criterion	1.773399
Sum squared resid	34.59049	Schwarz criterion	1.872737
Log likelihood	-91.76352	F-statistic	1.313929
Durbin-Watson stat	1.498132	Prob(F-statistic)	0.273884

Note that if appropriate, the alternative specification will be estimated using the cross-section or period GLS weights obtained from the restricted specification. If these weights were not saved with the restricted specification and are not available, you may first be asked to reestimate the original specification.

Redundant Variables Test

You may perform an F-est of the joint significance of variables that are presently included in a panel or pool equation estimated by list. Select **View/Coefficient Tests/Redundant Variables - Likelihood Ratio...** and in the resulting dialog, enter the names of the variables in the current specification that you wish to remove in the restricted model.

When you click on **OK**, EViews will estimate the restricted specification, form the usual F-est, and will display the test results and restricted estimates. Note that if appropriate, the alternative specification will be estimated using the cross-section or period GLS weights obtained from the unrestricted specification. If these weights were not saved with the specification and are not available, you may first be asked to reestimate the original specification.

To illustrate the redundant variables test, consider Example 10.4 from Wooldridge (2002, p. 262), where we test for the redundancy of GRANT and GRANT_1 in a specification estimated with cross-section random effects. The top portion of the unrestricted specification is given by:

Dependent Variable: LSCRAP
Method: Panel EGLS (Cross-section random effects)
Date: 11/24/04 Time: 11:25
Sample: 1987 1989
Cross-sections included: 54
Total panel (balanced) observations: 162
Swamy and Arora estimator of component variances

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.414833	0.242965	1.707379	0.0897
D88	-0.093452	0.108946	-0.857779	0.3923
D89	-0.269834	0.131397	-2.053577	0.0417
UNION	0.547802	0.409837	1.336635	0.1833
GRANT	-0.214696	0.147500	-1.455565	0.1475
GRANT_1	-0.377070	0.204957	-1.839747	0.0677

Effects Specification		
	S.D.	Rho
Cross-section random	1.390029	0.8863
Idiosyncratic random	0.497744	0.1137

Note in particular that our unrestricted model is a random effects specification using Swamy and Arora estimators for the component variances, and that the estimates of the cross-section and idiosyncratic random effects standard deviations are 1.390 and 0.4978, respectively.

If we select the redundant variables test, and perform a joint test on GRANT and GRANT_1, EViews displays the test results in the top of the results window:

Redundant Variables: GRANT GRANT_1

F-statistic	1.832264	Prob. F(2,156)	0.163478

Here we see that the statistic value of 1.832 does not lead us to reject, at conventional significant levels, the null hypothesis that GRANT and GRANT_1 are redundant in the unrestricted specification.

The restricted test equation results are depicted in the bottom portion of the window. Here we see the top portion of the results for the restricted equation:

Test Equation:
Dependent Variable: LSCRAP
Method: Panel EGLS (Cross-section random effects)
Date: 11/24/04 Time: 11:31
Sample: 1987 1989
Cross-sections included: 54
Total panel (balanced) observations: 162
Use pre-specified random component estimates
Swamy and Arora estimator of component variances

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	0.419327	0.073162	5.731525	0.0000
D88	-0.168993	0.095791	-1.764187	0.0796
D89	-0.442265	0.095791	-4.616981	0.0000
UNION	0.534321	0.082957	6.440911	0.0000

Effects Specification		
	S.D.	Rho
Cross-section random	1.390029	0.8863
Idiosyncratic random	0.497744	0.1137

The first thing to note is that the restricted specification removes the test variables GRANT and GRANT_1. Note further that the output indicates that we are using existing estimates of the random component variances ("Use pre-specified random component estimates"), and that the displayed results for the effects match those for the unrestricted specification.

Fixed Effects Testing

EViews 5.1 provides built-in tools for testing the joint significance of the fixed effects estimates in least squares specifications. To test the significance of your effects you must first estimate the unrestricted specification that includes the effects of interest. Next, select **View/Fixed/Random Effects Testing/Redundant Fixed Effects – Likelihood Ratio**. EViews will estimate the appropriate restricted specifications, and will display the test output as well as the results for the restricted specifications.

Note that where the unrestricted specification is a two-way fixed effects estimator, EViews will test the joint significance of all of the effects as well as the joint significance of the cross-section effects and the period effects separately.

Let us consider Example 3.6.2 in Baltagi (2001), in which we estimate a two-way fixed effects model. The results for the unrestricted estimated gasoline demand equation are given by:

Dependent Variable: LGASPCAR
Method: Panel Least Squares
Date: 11/24/04 Time: 11:57
Sample: 1960 1978
Cross-sections included: 18
Total panel (balanced) observations: 342

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-0.855103	0.385169	-2.220073	0.0272
LINCOMEP	0.051369	0.091386	0.562103	0.5745
LRPMG	-0.192850	0.042860	-4.499545	0.0000
LCARPCAP	-0.593448	0.027669	-21.44787	0.0000

Effects Specification

Cross-section fixed (dummy variables)
Period fixed (dummy variables)

R-squared	0.980564	Mean dependent var	4.296242
Adjusted R-squared	0.978126	S.D. dependent var	0.548907
S.E. of regression	0.081183	Akaike info criterion	-2.077237
Sum squared resid	1.996961	Schwarz criterion	-1.639934
Log likelihood	394.2075	F-statistic	402.2697
Durbin-Watson stat	0.348394	Prob(F-statistic)	0.000000

Note that the specification has both cross-section and period fixed effects. When you select the fixed effect test from the equation menu, EViews estimates three restricted specifications: one with period fixed effects only, one with cross-section fixed effects only, and one with only a common intercept. The test results are displayed at the top of the results window:

Redundant Fixed Effects Tests
Equation: Untitled
Test cross-section and period fixed effects

Effects Test	Statistic	d.f.	Prob.
Cross-section F	113.351303	(17,303)	0.0000
Cross-section Chi-square	682.635958	17	0.0000
Period F	6.233849	(18,303)	0.0000
Period Chi-square	107.747064	18	0.0000
Cross-Section/Period F	55.955615	(35,303)	0.0000
Cross-Section/Period Chi-square	687.429282	35	0.0000

Notice that there are three sets of tests. The first set consists of two tests that evaluate the joint significance of the cross-section effects using sums-of-squares (F-est) and the likelihood function (Chi-square test). The corresponding restricted specification is one in which there are period effects only. The two statistic values (113.35 and 682.64) and the associated p-values strongly reject the null that the effects are redundant.

The remaining results evaluate the joint significance of the period effects, and of all of the effects, respectively. All of the results suggest that the corresponding effects are statistically significant.

Below the test statistic results, EViews displays the results for the test equations. In this example, there are three distinct restricted equations so EViews shows three sets of estimates.

Lastly, note that this test statistic is not currently available for instrumental variables and GMM specifications.

Hausman Test for Correlated Random Effects

A central assumption in random effects estimation is the assumption that the random effects are uncorrelated with the explanatory variables. One common method for testing this assumption is to employ a Hausman (1978) test to compare the fixed and random effects estimates of coefficients (for discussion see, for example Wooldridge (2002, p. 288), and Baltagi (2001, p. 65)).

To perform the Hausman test, you must first estimate a model with your random effects specification. Next, select **View/Fixed/Random Effects Testing/Correlated Random Effects - Hausman Test**. EViews will automatically estimate the corresponding fixed effects specifications, compute the test statistics, and display the results and auxiliary equations.

For example, Baltagi (2001) considers an example of Hausman testing (Example 1, p. 69), in which the results for a Swamy-Arora random effects estimator for the Grunfeld data are compared with those obtained from the corresponding fixed effects estimator. To perform this test in EViews 5.1, we first estimate the random effects estimator, obtaining the results:

Dependent Variable: I
Method: Panel EGLS (Cross-section random effects)
Date: 11/24/04 Time: 12:45
Sample: 1935 1954
Cross-sections included: 10
Total panel (balanced) observations: 200
Swamy and Arora estimator of component variances

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-57.83441	28.88930	-2.001932	0.0467
F	0.109781	0.010489	10.46615	0.0000
K	0.308113	0.017175	17.93989	0.0000

Effects Specification	S.D.	Rho
Cross-section random	84.20095	0.7180
Idiosyncratic random	52.76797	0.2820

Next we select the Hausman test from the equation menu by clicking on **View/Fixed/Random Effects Testing/Hausman Test of Random vs. Fixed**. EViews estimates the corresponding fixed effects estimator, evaluates the test, and displays the results in the equation window. If the original specification is a two-way random effects model, EViews will test the two sets of effects separately as well as jointly.

There are three parts to the output. The top portion describes the test statistic and provides a summary of the results. Here we have:

Hausman Specification Test (Random vs. Fixed Effects)
Equation: EQ263
Test for correlated cross-section random effects

Test Summary	Chi-Sq. Statistic	Chi-Sq. d.f.	Prob.
Cross-section random	2.131366	2	0.3445

The statistic provide little evidence against the null hypothesis that there is no misspecification.

The next portion of output provides additional test detail, showing the coefficient estimates from both the random and fixed effects estimators, along with the variance of the difference and associated p-values for the hypothesis that there is no difference. Note that in some cases, the estimated variances can be negative so that the probabilities cannot be computed.

Cross-section random effects test comparisons:

Variable	Fixed	Random	Var(Diff.)	Prob.
F	0.110124	0.109781	0.000031	0.9506
K	0.310065	0.308113	0.000006	0.4332

The bottom portion of the output contains the results from the corresponding fixed effects estimation:

Cross-section random effects test equation:
Dependent Variable: I
Method: Panel Least Squares
Date: 11/24/04 Time: 12:51
Sample: 1935 1954
Cross-sections included: 10
Total panel (balanced) observations: 200

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-58.74394	12.45369	-4.716990	0.0000
F	0.110124	0.011857	9.287901	0.0000
K	0.310065	0.017355	17.86656	0.0000

Effects Specification
Cross-section fixed (dummy variables)

In some cases, EViews will automatically drop non-varying variables in order to construct the test statistic. These dropped variables will be indicated in this latter estimation output.

Estimation Background

The basic class of models that can be estimated using panel techniques may be written as:

$$Y_{it} = f(X_{it}, \beta) + \delta_i + \gamma_t + \epsilon_{it} \tag{29.1}$$

The leading case involves a linear conditional mean specification, so that we have:

$$Y_{it} = \alpha + X_{it}'\beta + \delta_i + \gamma_t + \epsilon_{it} \tag{29.2}$$

where Y_{it} is the dependent variable, and X_{it} is a k-vector of regressors, and ϵ_{it} are the error terms for $i = 1, 2, \ldots, M$ cross-sectional units observed for dated periods $t = 1, 2, \ldots, T$. The α parameter represents the overall constant in the model, while the δ_i and γ_t represent cross-section or period specific effects (random or fixed).

Note that in contrast to the pool specifications described in Equation (27.2) on page 860, EViews panel equations allow you to specify equations in general form, allowing for non-linear coefficients mean equations with additive effects. Panel equations do not automatically allow for β coefficients that vary across cross-sections or periods, but you may, of course, create interaction variables that permit such variation.

Other than these differences, the pool equation discussion of "Estimation Background" on page 859 applies to the estimation of panel equations. In particular, the calculation of fixed and random effects, GLS weighting, AR estimation, and coefficient covariances for least squares and instrumental variables is equally applicable in the present setting.

Accordingly, the remainder of this discussion will focus on a brief review of the relevant econometric concepts surrounding GMM estimation of panel equations.

GMM Details

The following is a brief review of GMM estimation and dynamic panel estimators. As always, the discussion is merely an overview. For detailed surveys of the literature, see Wooldridge (2002) and Baltagi (2001).

Background

The basic GMM panel estimators are based on moments of the form,

$$g(\beta) = \sum_{i=1}^{M} g_i(\beta) = \sum_{i=1}^{M} Z_i' \epsilon_i(\beta) \tag{29.3}$$

where Z_i is a $T_i \times p$ matrix of instruments for cross-section i, and,

$$\epsilon_i(\beta) = (Y_i - f(X_{it}, \beta)) \tag{29.4}$$

In some cases we will work symmetrically with moments where the summation is taken over periods t instead of i.

GMM estimation minimizes the quadratic form:

$$\begin{aligned} S(\beta) &= \left(\sum_{i=1}^{M} Z_i' \epsilon_i(\beta) \right)' H \left(\sum_{i=1}^{M} Z_i' \epsilon_i(\beta) \right) \\ &= g(\beta)' H g(\beta) \end{aligned} \tag{29.5}$$

with respect to β for a suitably chosen $p \times p$ weighting matrix H.

Given estimates of the coefficient vector, $\hat{\beta}$, an estimate of the coefficient covariance matrix is computed as,

$$V(\hat{\beta}) = (G'HG)^{-1}(G'H\Lambda HG)(G'HG)^{-1} \tag{29.6}$$

where Λ is an estimator of $E(g_i(\beta)g_i(\beta)') = E(Z_i'\epsilon_i(\beta)\epsilon_i(\beta)'Z_i)$, and G is a $T_i \times k$ derivative matrix given by:

$$G(\beta) = \left(-\sum_{i=1}^{M} Z_i'\nabla f_i(\beta)\right) \tag{29.7}$$

In the simple linear case where $f(X_{it}, \beta) = X_{it}'\beta$, we may write the coefficient estimator in closed form as,

$$\begin{aligned}\hat{\beta} &= \left(\left(\sum_{i=1}^{M} Z_i'X_i\right)' H \left(\sum_{i=1}^{M} Z_i'X_i\right)\right)^{-1}\left(\left(\sum_{i=1}^{M} Z_i'X_i\right)' H \left(\sum_{i=1}^{M} Z_i'Y_i\right)\right) \\ &= (M_{ZX}'HM_{ZX})^{-1}(M_{ZX}'HM_{ZY})\end{aligned} \tag{29.8}$$

with variance estimator,

$$V(\hat{\beta}) = (M_{ZX}'HM_{ZX})^{-1}(M_{ZX}'H\Lambda HM_{ZX})(M_{ZX}'HM_{ZX})^{-1} \tag{29.9}$$

for M_{AB} of the general form:

$$M_{AB} = M^{-1}\left(\sum_{i=1}^{M} A_i'B_i\right) \tag{29.10}$$

The basics of GMM estimation involve: (1) specifying the instruments Z, (2) choosing the weighting matrix H, and (3) determining an estimator for Λ.

It is worth pointing out that the summations here are taken over individuals; we may equivalently write the expressions in terms of summations taken over periods. This symmetry will prove useful in describing some of the GMM specifications that EViews supports.

A wide range of specifications may be viewed as specific cases in the GMM framework. For example, the simple 2SLS estimator, using ordinary estimates of the coefficient covariance, specifies:

$$\begin{aligned}H &= (\hat{\sigma}^2 M_{ZZ})^{-1} \\ \Lambda &= \hat{\sigma}^2 M_{ZZ}\end{aligned} \tag{29.11}$$

Substituting, we have the familiar expressions,

$$\begin{aligned}\hat{\beta} &= (M_{ZX}'(\hat{\sigma}^2 M_{ZZ})^{-1}M_{ZX})^{-1}(M_{ZX}'(\hat{\sigma}^2 M_{ZZ})^{-1}M_{ZY}) \\ &= (M_{ZX}'M_{ZZ}^{-1}M_{ZX})^{-1}(M_{ZX}'M_{ZZ}^{-1}M_{ZY})\end{aligned} \tag{29.12}$$

and,

$$V(\hat{\beta}) = \hat{\sigma}^2 (M_{ZX}' M_{ZZ}^{-1} M_{ZX})^{-1} \tag{29.13}$$

Standard errors that are robust to conditional or unconditional heteroskedasticity and contemporaneous correlation may be computed by substituting a new expression for Λ,

$$\Lambda = T^{-1}\left(\sum_{t=1}^{T} Z_t' \hat{\epsilon}_t \hat{\epsilon}_t' Z_t \right) \tag{29.14}$$

so that we have a White cross-section robust coefficient covariance estimator. Additional robust covariance methods are described in detail in "Robust Coefficient Covariances" on page 869.

In addition, EViews supports a variety of weighting matrix choices. All of the choices available for covariance calculation are also available for weight calculations in the standard panel GMM setting: 2SLS, White cross-section, White period, White diagonal, Cross-section SUR (3SLS), Cross-section weights, Period SUR, Period weights. An additional differenced error weighting matrix may be employed when estimating a dynamic panel data specification using GMM.

The formulae for these weights are follow immediately from the choices given in "Robust Coefficient Covariances" on page 869. For example, the Cross-section SUR (3SLS) weighting matrix is computed as:

$$H = \left(T^{-1} \sum_{t=1}^{T} Z_t' \hat{\Omega}_M Z_t \right)^{-1} \tag{29.15}$$

where $\hat{\Omega}_M$ is an estimator of the contemporaneous covariance matrix. Similarly, the White period weights are given by:

$$H = \left(M^{-1} \sum_{i=1}^{M} Z_i' \hat{\epsilon}_i \hat{\epsilon}_i' Z_i \right)^{-1} \tag{29.16}$$

These latter GMM weights are associated with specifications that have arbitrary serial correlation and time-varying variances in the disturbances.

GLS Specifications

EViews allows you to estimate a GMM specification on GLS transformed data. Note that the moment conditions are modified to reflect the GLS weighting:

$$g(\beta) = \sum_{i=1}^{M} g_i(\beta) = \sum_{i=1}^{M} Z_i' \hat{\Omega}^{-1} \epsilon_i(\beta) \tag{29.17}$$

Dynamic Panel Data

Consider the linear dynamic panel data specification given by:

$$Y_{it} = \sum_{j=1}^{p} \rho_j Y_{it-j} + X_{it}'\beta + \delta_i + \epsilon_{it} \tag{29.18}$$

First-differencing this specification eliminates the individual effect and produces an equation of the form:

$$\Delta Y_{it} = \sum_{j=1}^{p} \rho_j \Delta Y_{it-j} + \Delta X_{it}'\beta + \Delta\epsilon_{it} \tag{29.19}$$

which may be estimated using GMM techniques.

Efficient GMM estimation of this equation will typically employ a different number of instruments for each period, with the period-specific instruments corresponding to the different numbers of lagged dependent and predetermined variables available at a given period. Thus, along with any strictly exogenous variables, one may use period-specific sets of instruments corresponding to lagged values of the dependent and other predetermined variables.

Consider, for example, the motivation behind the use of the lagged values of the dependent variable as instruments in Equation (29.19). If the innovations in the original equation are i.i.d., then in $t = 3$, the first period available for analysis of the specification, it is obvious that Y_{i1} is a valid instrument since it is correlated with ΔY_{i2}, but uncorrelated with $\Delta\epsilon_{i3}$. Similarly, in $t = 4$, both Y_{i2} and Y_{i1} are potential instruments. Continuing in this vein, we may form a set of predetermined instruments for individual i using lags of the dependent variable:

$$W_i = \begin{bmatrix} Y_{i1} & 0 & 0 & \dots & \dots & \dots & \dots & 0 \\ 0 & Y_{i1} & Y_{i2} & \dots & \dots & \dots & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & Y_{i1} & Y_{i2} & \dots & Y_{iT_i-2} \end{bmatrix} \tag{29.20}$$

Similar sets of instruments may be formed for each predetermined variables.

Assuming that the ϵ_{it} are not autocorrelated, the optimal GMM weighting matrix for the differenced specification is given by,

$$H^d = \left(M^{-1} \sum_{i=1}^{M} Z_i'\Xi Z_i \right)^{-1} \tag{29.21}$$

where Ξ is the matrix,

$$\Xi = \frac{1}{2}\begin{bmatrix} 2 & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & 0 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 2 & -1 \\ 0 & 0 & 0 & \dots & -1 & 2 \end{bmatrix}\sigma^2 \tag{29.22}$$

and where Z_i contains a mixture of strictly exogenous and predetermined instruments. Note that this weighting matrix is the one used in the one-step Arellano-Bond estimator.

Given estimates of the residuals from the one-step estimator, we may replace the H^d weighting matrix with one estimated using computational forms familiar from White period covariance estimation:

$$H = \left(M^{-1}\sum_{i=1}^{M} Z_i'\Delta\epsilon_i\Delta\epsilon_i'Z_i\right)^{-1} \tag{29.23}$$

This weighting matrix is the one used in the Arellano-Bond two-step estimator.

Lastly, we note that an alternative method of transforming the original equation to eliminate the individual effect involves computing orthogonal deviations (Arellano and Bover, 1995). We will not reproduce the details on here but do note that residuals transformed using orthogonal deviations have the property that the optimal first-stage weighting matrix for the transformed specification is simply the 2SLS weighting matrix:

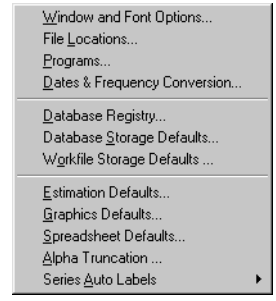
$$H = \left(M^{-1}\sum_{i=1}^{M} Z_i'Z_i\right)^{-1} \tag{29.24}$$

Appendix A. Global Options

EViews employs user-specified default settings in many operations. You may, for example, set defaults for everything from how to perform frequency conversion between workfile pages of different frequency, to which font to use in table output, to line color and thickness in graphs, to how to compute derivatives in nonlinear estimation routines.

These default options may, of course, be overridden when actually performing an operation. For example, you may have specified a default conversion of data from monthly to quarterly data by averaging observations, but may choose to use summing when performing a specific conversion. Similarly, you may instruct EViews to use the color red for the first line in newly created graphs, and then change the color in a specific graph.

The Options Menu



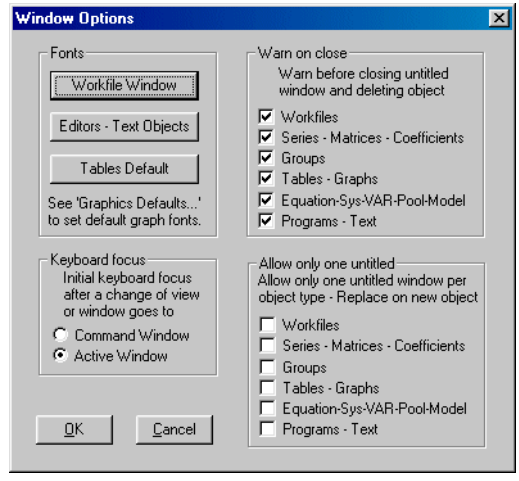
The **Options** menu in the main toolbar allows you to define the default behavior for many of the operations in EViews.

We discuss briefly each of these menu items. In some cases, additional detail is provided in the corresponding sections of the documentation.

Window and Font Options

The window and font options control the display characteristics of various types of EViews output. The settings are divided into broad groupings:

- The **Fonts** section in the upper left-hand corner of the dialog allows you to change the default font styles and sizes for various sets of windows and objects. Press the button corresponding to the type of object for which you want to change the font and select the new font in the **Font** dialog. For example, to set the default font face and size to be used in table objects and table views of objects, click on **Tables Default**, and select the font.



- **Keyboard Focus** controls where the keyboard cursor is placed when you change views or windows. As the label suggests, when you select **Command Window**, the keyboard focus will go to the command window following a change of view. This setting is most useful if you primarily use EViews via the command line. Choosing **Active Window** will cause the focus to go to the active window following a change of view. You will find this setting useful if you wish to use keystrokes to navigate between and to select items in various windows. Note that whatever the default setting, you may always change the keyboard focus by clicking in the command window, or by clicking in the active window.

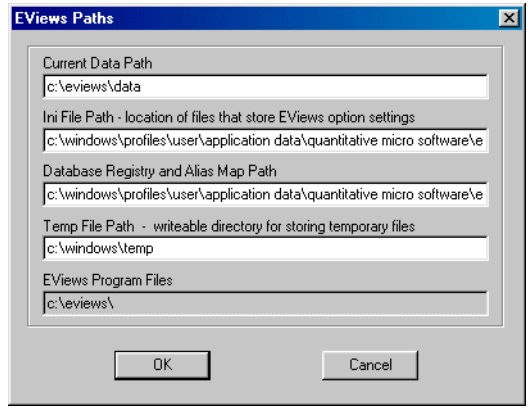
- **Warn On Close** instructs EViews to provide a warning message when you close an untitled object. You may choose to set warnings for various object types. By default, EViews warns you that closing an unnamed object without naming it will cause the object to be deleted. Along with the warning, you will be given an opportunity to name the object. If you turn the warning off, closing an untitled window will automatically delete the object.

- **Allow Only One Untitled** specifies, for each object type, whether to allow multiple untitled objects to be opened at the same time. If only one is allowed, creating a new untitled object will cause any existing untitled object of that type to be deleted automatically. Setting EViews to allow only one untitled object reduces the number of windows that you will see on your desktop.

  If you elect to allow only one untitled object, we strongly recommend that you select **Warn on Close**. With this option set, you will be given the opportunity to name and save an existing untitled object, otherwise the object will simply be deleted.

File Locations

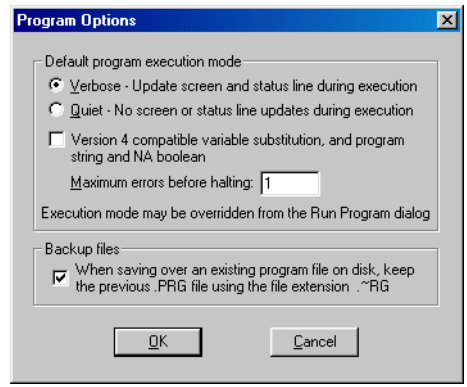
This dialog allows you to set the default working directory, and the locations of the .INI file, database registry and alias map, and the temp directory.



The dialog also reports, but does not allow you to alter, the location of your EViews executable.

Note that the default working directory may also be changed via the **File Open** and **File Save** or **File Save As** dialogs, or by using the `cd` command.

Programs


Program Options
Default program execution mode
Verbose - Update screen and status line during execution
Quiet - No screen or status line updates during execution
Version 4 compatible variable substitution, and program string and NA boolean
Maximum errors before halting: 1
Execution mode may be overridden from the Run Program dialog
Backup files
When saving over an existing program file on disk, keep the previous .PRG file using the file extension .~RG
OK
Cancel

The **Program Options** dialog specifies whether, by default, EViews runs programs in **Verbose** mode, listing commands in the status line as they are executed, or whether it uses **Quiet** mode, which suppresses this information. EViews will run faster in quiet mode since the status line display does not need to be updated.

This default may always be overridden from the **Run Program** dialog, or by using the option "q" in the run statement, as in:

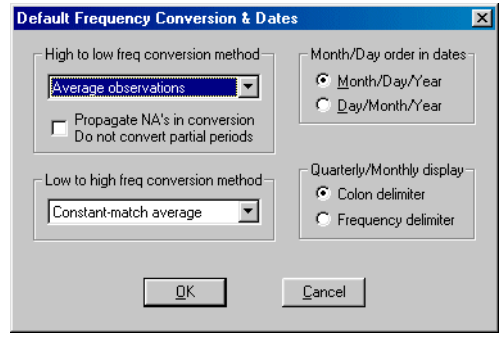
```
run(q) myprogram
```

For details, see Chapter 6, "EViews Programming", on page 83 of the *Command and Programming Reference*.

If the checkbox labeled **Version 4 compatible variable substitution** is selected, EViews will use the variable substitution behavior found in EViews 4 and earlier versions. EViews 5 has changed the way that substitution variables are evaluated in expressions. You may use this checkbox to use Version 4 substitution rules. See "Version 5 Compatibility Notes" on page 93 of the *Command and Programming Reference* for additional discussion.

In addition, you may use this dialog to specify the maximum number of errors before halting program execution, and whether EViews should keep backup copies of program files when saving over an existing file. The backup copy will have the same name as the file, but with the first character in the extension changed to " ~ ".

Dates & Frequency Conversion



This dialog allows you to set the default frequency conversion methods for both up and down conversion, and the default method of displaying dates.

The default frequency conversion method tells EViews how to convert data when you move data to lower or higher frequency workfiles. The frequency conversion methods and use of default set-

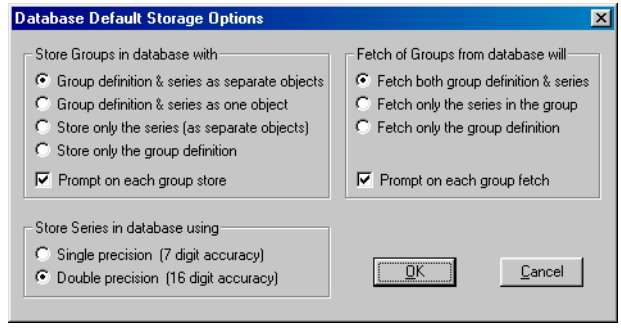
tings are discussed in detail in "Frequency Conversion" beginning on page 115.

The default date display controls the format for dates in sample processing, and in workfile and various object views. For daily and weekly data, you can set the default to American (**Month/Day/Year**) or you may switch to European notation (**Day/Month/Year**), where the day precedes the month. You may also specify your quarterly or monthly date display to use the **Colon delimiter** or a **Frequency delimiter** character based on the workfile frequency ("q" or "m"). The latter has the advantage of displaying dates using an informative delimiter ("1990q1" vs. "1990:1").

See also "Free-format Conversion Details" on page 149 for related discussion.

Database Registry / Database Storage Defaults

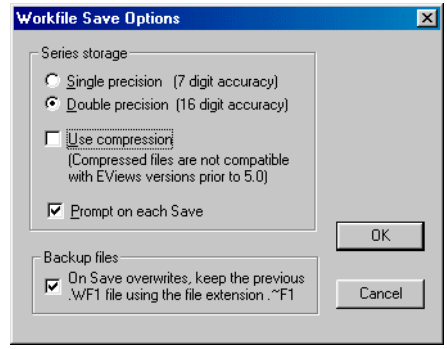
The **Database Registry...** settings are described in detail in "The Database Registry" on page 275.


Database Default Storage Options
Store Groups in database with
Group definition & series as separate objects
Group definition & series as one object
Store only the series (as separate objects)
Store only the group definition
Prompt on each group store
Fetch of Groups from database will
Fetch both group definition & series
Fetch only the series in the group
Fetch only the group definition
Prompt on each group fetch
Store Series in database using
Single precision (7 digit accuracy)
Double precision (16 digit accuracy)
OK
Cancel

You may also control the default behavior of EViews when moving data into and out of databases using the **Database Storage Defaults...** menu item to open the **Database Default Storage Options** dialog. The dialog controls the behavior of group store and fetch (see "Storing a Group Object" and "Fetching a Group Object" on page 273), and whether data are stored to databases in single or double precision.

Workfile Storage Defaults

The **Workfile Save Options** dialog allows you to specify storage precision and compression options for your saved workfiles, and to specify whether or not to create backup workfiles when overwriting existing files on disk.


Workfile Save Options
Series storage
Single precision (7 digit accuracy)
Double precision (16 digit accuracy)
Use compression
(Compressed files are not compatible with EViews versions prior to 5.0)
Prompt on each Save
OK
Backup files
On Save overwrites, keep the previous .WF1 file using the file extension .~F1
Cancel

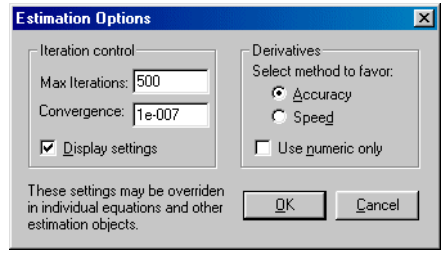
The **Series storage** portion of the dialog controls whether series data are stored in saved workfiles in single or double precision. Note that workfiles saved in single precision will be converted to double precision when they are loaded into memory.

In addition, you may elect to save your workfiles in compressed format on disk by checking the **Use compression** setting. Note that compressed files are not readable by versions of EViews prior to 5.0, and that they do not save memory when using the workfile, as it is uncompressed when loaded into memory. Compressed workfiles do, however, save disk space.

You should uncheck the **Prompt on each Save** checkbox to suppress the workfile save dialog on each workfile save.

In addition, you may specify whether EViews should keep backup copies of workfiles when saving over an existing file. If selected, the automatic backup copy will have the same name as the file, but with the first character in the extension changed to " ~ ".

Estimation Defaults



You can set the global defaults for the maximum number of iterations, convergence criterion, and methods for computing derivatives.

These settings will be used as the default settings in iterative estimation methods for equation, log likelihood, state space, and system objects. Note that previously estimated EViews objects that were estimated with the previous default settings will be unchanged, unless reestimated. See "Setting Estimation Options" on page 951 for additional details.

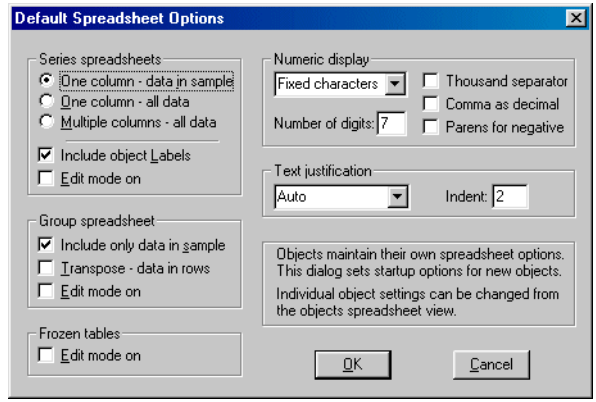
When the **Display settings** option is checked, EViews will produce additional output in the estimation output describing the settings under which estimation was performed.

Graphics Defaults

These options control how a graph appears when it is first created. Most of these options are explained in considerable detail in "Modifying Graphs" beginning on page 416.

Spreadsheet Defaults

These options control the default spreadsheet view settings of series, group, and table objects. The defaults are set using these option settings when the object is first created.



The left-hand portion of the dialog contains settings for specific objects.

In the upper left-hand corner of the dialog, you may set the display format for **Series spreadsheets**. You may choose to display the spreadsheet in one or multiple columns, with or without object labels and header information. In addition, you may choose to have edit mode turned on or off by default.

Below this section are the **Group spreadsheet** options for sample filtering, transposed display, and edit mode.

Frozen tables allows you to modify whether edit mode is on or off by default.

On the right-hand side of the dialog are display settings that apply to all numeric and text spreadsheet displays. You may specify the default **Numeric display** to be used by new objects. The settings allow you to alter the numeric formatting of cells, the number of digits to be displayed, as well as the characters used as separators and indicators for negative numbers. Similarly, you may choose the default **Text justification** and **Indentation** for alphanumeric display.

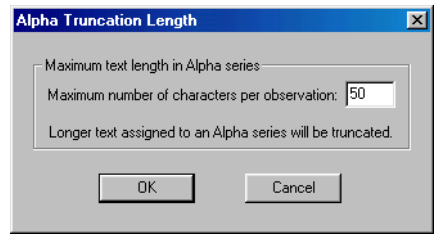
We emphasize the fact that these display options only apply to newly created series or group objects. If you subsequently alter the defaults, existing objects will continue to use their own settings.

Alpha Truncation

Note that EViews alpha series automatically resize as needed, up to the truncation length.

To modify the alpha series truncation length, select **Alpha Truncation...** to open the **Alpha Truncation Length** dialog, and enter the desired length. Subsequent alpha series creation and assignment will use the new truncation length.

You should bear in mind that the strings in EViews alpha series are of fixed length, so that the size of each observation is equal to the length of the longest string. If you have a series with all short strings, with the exception of one very long string, the memory taken up by the series will be the number of observations times the longest string size.


Alpha Truncation Length
Maximum text length in Alpha series
Maximum number of characters per observation:
50
Longer text assigned to an Alpha series will be truncated.
OK
Cancel

Series Auto Labels

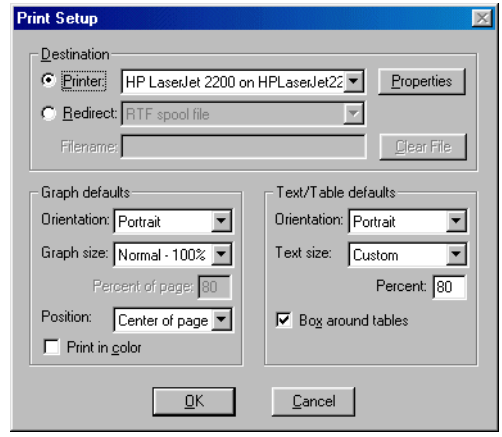
You may elect to have EViews keep a history of the commands that created or modified a series as part of the series label. You can choose to turn this option on or off.

Note that the series label may be viewed by selecting **View/Label** in a series window, or at the top of a series spreadsheet if the spreadsheet defaults are set to display labels ("Spreadsheet Defaults" on page 942).

Print Setup

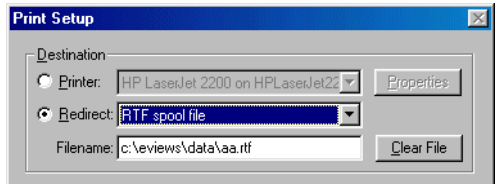
The **Print Setup** options (**File/Print Setup...** on the main menu) determine the default print behavior when you print an object view.

The top of the **Print Setup** dialog provides you with choices for the destination of printed output.



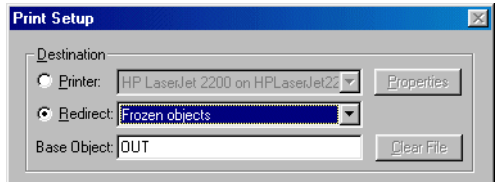
You may elect to send your output directly to the printer by selecting the **Printer** radio button. The drop down menu allows you to choose between the printer instances available on your computer, and the **Properties** button allows you to customize settings for the selected printer.

Alternatively, you may select **Redirect** so that EViews will redirect your output. There are three possible settings in the drop down menu: **RTF spool file**, **Text spool file (graphs print)**, and **Frozen objects**.

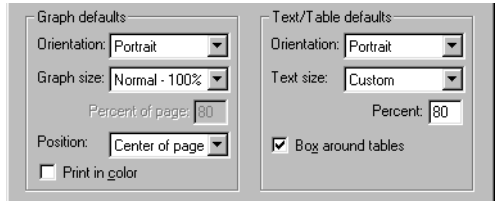


If you select **RTF spool file**, all of your subsequent print commands will be appended to the RTF file specified in **Filename** (the file will be created in the default path with an ".RTF" extension). You may use the **Clear File** button to empty the contents of the existing file.

If you select **Text spool file (graphs print)**, all printed text output will be appended to the text file specified in **Filename** (the file will be created in the default path with a ".TXT" extension). Since graph output cannot be saved in text format, all printing involving graphs will be sent to the printer.



The last selection, **Frozen objects**, redirects print commands into newly created graph or table objects using the specified **Base Object** name. Each subsequent print command will create a new table or graph object in the current workfile, naming it using the base name and an identifying number. For example, if you supply the base name of "OUT", the first print command will generate a table or graph named OUT01, the second print command will generate OUT02, and so on.



The bottom portion of the **Print Setup** dialog sets various default settings for graph and table printing.

The **Graph defaults** section has settings for printing in portrait or landscape mode, scaling the size of the graph, positioning the graph on the page, and choosing black and white or color printing. Note that some graph settings are not relevant when redirecting output, so that portions of the dialog may be grayed out.

The **Text/Table defaults** allow you to print in portrait or landscape mode, scale the size of the text, and to draw or not draw a box around the table. Some of these options are not available when redirecting output.

Note that these settings only specify the default settings for printing. You may override any of these settings when printing using the **Print** dialog or `print` command.

Appendix B. Wildcards

EViews supports the use of wildcard characters in a variety of situations where you need to enter a list of objects or a list of series. For example, you can use wildcards to:

- fetch, store, copy, rename or delete a list of objects
- specify a group object
- query a database by name or filter the workfile display

The following discussion describes some of the issues involved in the use of wildcard characters and expressions.

Wildcard Expressions

There are two wildcard characters: "*" and "?". The wildcard character "*" matches zero or more characters in a name, and the wildcard "?" matches any single character in a name.

For example, you can use the wildcard expression "GD*" to refer to all objects whose names begin with the characters "GD". The series GD, GDP, GD_F will be included in this list GGD, GPD will not. If you use the expression GD?, EViews will interpret this as a list of all objects with three character names beginning with the string "GD": GDP and GD2 will be included, but GD, GD_2 will not.

You can instruct EViews to match a fixed number of characters by using as many "`?`" wildcard characters as necessary. For example, EViews will interpret "??GDP" as matching all objects with names that begin with any two characters followed by the string "GDP". USGDP and F_GDP will be included but GDP, GGDP, GDPUS will not.

You can also mix the different wildcard characters in an expression. For example, you can use the expression "*GDP?" to refer to any object that ends with the string "GDP" and an arbitrary character. Both GDP_1, USGDP_F will be included.

Using Wildcard Expressions

Wildcard expressions may be used in filtering the workfile display (see "Filtering the Workfile Directory Display" on page 59), in selected EViews commands, and in creating a group object.

The following commands support the use of wildcards: `show`, `store`, `fetch`, `copy`, `rename` and `delete`.

To create a group using wildcards, simply select **Object/New Object.../Group**, and enter the expression, EViews will first expand the expression, and then attempt to create a group using the corresponding list of series. For example, entering the list,

```
y x*
```

will create a group comprised of Y and all series beginning with the letter X. Alternatively, you can enter the command:

```
group g1 x* y?? c
```

defines a group G1, consisting of all of the series matching "X*", and all series beginning with the letter "Y" followed by two arbitrary characters.

When making a group, EViews will only select series objects which match the given name pattern and will place these objects in the group.

Once created, these groups may be used anywhere that EViews takes a group as input. For example, if you have a series of dummy variables, DUM1, DUM2, DUM3, ..., DUM9, that you wish to enter in a regression, you can create a group containing the dummy variables, and then enter the group in the regression:

```
group gdum dum?
equation eq1.ls y x z gdum
```

will run the appropriate regression. Note that we are assuming that the dummy variables are the only series objects which match the wildcard expression DUM?.

Source and Destination Patterns

When wildcards are used during copy and rename operations, a pattern must be provided for both the source and the destination. The destination pattern must always conform to the source pattern in that the number and order of wildcard characters must be exactly the same between the two. For example, the patterns,

Source Pattern	Destination Pattern
x*	y*
*c	b*
x*12?	yz*f?abc

which conform to each other, while these patterns do not:

Source Pattern	Destination Pattern
a*	b

*x	?y
x*y*	*x*y*

When using wildcards, the new destination name is formed by replacing each wildcard in the destination pattern by the characters from the source name that matched the corresponding wildcard in the source pattern. This allows you to both add and remove characters from the source name during the copy or rename process. Some examples should make this clear:

Source Pattern	Destination Pattern	Source Name	Destination Name
*_base	*_jan	x_base	x_jan
us_*	*	us_gdp	gdp
x?	x?f	x1	x1f
_	**f	us_gdp	usgdpf
??*f	??_*	usgdpf	us_gdp

Note, as shown in the second example, that a simple asterisk for the destination pattern will result in characters being removed from the source name when forming the destination name. To copy objects between containers preserving the existing name, either repeat the source pattern as the destination pattern,

```
copy x* db1::x*
```

or omit the destination pattern entirely,

```
copy x* db1::
```

Resolving Ambiguities

Note that an ambiguity can arise with wildcard characters since both "*" and "?" have multiple uses. The "*" character may be interpreted as either a multiplication operator or a wildcard character. The "?" character serves as both the single character wildcard and the pool cross section identifier.

Wildcard versus Multiplication

There is a potential for ambiguity in the use of the wildcard character "*".

Suppose you have a workfile with the series X, X2, Y, XYA, XY2. There are then two interpretations of the wildcard expression "X*2". The expression may be interpreted as an auto-series representing X multiplied by 2. Alternatively, the expression may be used as a wildcard expression, referring to the series X2 and XY2.

Note that there is only an ambiguity when the character is used in the middle of an expression, not when the wildcard character "*" is used at the beginning or end of an expression. EViews uses the following rules to determine the interpretation of ambiguous expressions:

- EViews first tests to see whether the expression represents a valid series expression. If so, the expression is treated as an auto-series. If it is not a valid series expression, then EViews will treat the "*" as a wildcard character. For example,

  ```
  y*x
  2*x
  ```

  are interpreted as auto-series, while,

  ```
  *x
  x*a
  ```

  are interpreted as wildcard expressions.

- You can force EViews to treat "*" as a wildcard by preceding the character with another "*". Thus, expressions containing "**" will always be treated as wildcard expressions. For example, the expression:

  ```
  x**2
  ```

  unambiguously refers to all objects with names beginning with "X" and ending with "2". Note that the use of "**" does *not* conflict with the EViews exponentiation operator "^".

- You can instruct EViews to treat "*" as a series expression operator by enclosing the expression (or any subexpression) in parentheses. For example:

  ```
  (y*x)
  ```

  always refers to X times Y.

We *strongly* encourage you to resolve the ambiguity by using parentheses to denote series expressions, and double asterisks to denote wildcards (in the middle of expressions), whenever you create a group. This is especially true when group creation occurs in a program; otherwise the behavior of the program will be difficult to predict since it will change as the names of other objects in the workfile change.

Wildcard versus Pool Identifier

The "?" wildcard character is used both to match any single character in a pattern and as a place-holder for the cross-section identifier in pool objects.

EViews resolves this ambiguity by not allowing the wildcard interpretation of "?" in any expression involving a pool object or entered into a pool dialog. "?" is used exclusively as

a cross-section identifier. For example, suppose that you have the pool object POOL1. Then, the expression,

```
pool1.est y? x? c
```

is a regression of the pool variable Y? on the pool variable X?, and,

```
pool1.delete x?
```

deletes all of the series in the pool series X?. There is no ambiguity in the interpretation of these expressions since they both involve POOL1.

Similarly, when used apart from a pool object, the "?" is interpreted as a wildcard character. Thus,

```
delete x?
```

unambiguously deletes all of the series matching "X?".

Appendix C. Estimation and Solution Options

EViews estimates the parameters of a wide variety of nonlinear models, from nonlinear least squares equations, to maximum likelihood models, to GMM specifications. These types of nonlinear estimation problems do not have closed form solutions and must be estimated using iterative methods. EViews also solves systems of non-linear equations. Again, there are no closed form solutions to these problems, and EViews must use an iterative method to obtain a solution.

Below, we provide details on the algorithms used by EViews in dealing with nonlinear estimation and solution, and the optional settings that we provide to allow you to control estimation.

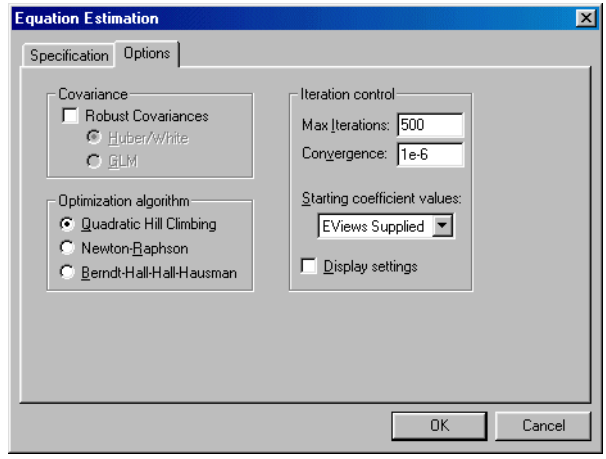
Our discussion here is necessarily brief. For additional details, we direct you to the quite readable discussions in Press, *et al.* (1992), Quandt (1983), Thisted (1988), and Amemiya (1983).

Setting Estimation Options

When you estimate an equation in EViews, you enter specification information into the **Specification** tab of the **Equation Estimation** dialog. Clicking on the **Options** tab displays a dialog that allows you to set various options to control the estimation procedure. The contents of the dialog will differ depending upon the options available for a particular estimation procedure.

The default settings for the options will be taken from the global options ("Estimation Defaults" on page 941), or from the options used previously to estimate the object.

The **Options** tab for binary models is depicted here. For other estimator and estimation techniques (*e.g.* systems) the dialog will differ to reflect the different estimation options that are available.



Starting Coefficient Values

Iterative estimation procedures require starting values for the coefficients of the model. There are no general

rules for selecting starting values for parameters. Obviously, the closer to the true values, the better, so if you have reasonable guesses for parameter values, these can be useful. In some cases, you can obtain starting values by estimating a restricted version of the model. In general, however, you may have to experiment to find good starting values.

EViews follows three basic rules for selecting starting values:

- For nonlinear least squares type problems, EViews uses the values in the coefficient vector at the time you begin the estimation procedure as starting values.
- For system estimators and ARCH, EViews uses starting values based upon preliminary single equation OLS or TSLS estimation. In the dialogs for these estimators, the drop-down menu for setting starting values will not appear.
- For selected estimation techniques (binary, ordered, count, censored and truncated), EViews has built-in algorithms for determining the starting values using specific information about the objective function. These will be labeled in the **Starting coefficient values** combo box as **EViews supplied**.

In the latter two cases, you may change this default behavior by selecting an item from the **Starting coefficient values** drop down menu. You may choose fractions of the default starting values, zero, or arbitrary **User Supplied**.

If you select **User Supplied**, EViews will use the values stored in the C coefficient vector at the time of estimation as starting values. To see the starting values, double click on the coefficient vector in the workfile directory. If the values appear to be reasonable, you can close the window and proceed with estimating your model.

If you wish to change the starting values, first make certain that the spreadsheet view of the coefficient vector is in edit mode, then enter the coefficient values. When you are finished setting the initial values, close the coefficient vector window and estimate your model.

You may also set starting coefficient values from the command window using the PARAM command. Simply enter the PARAM keyword, followed by pairs of coefficients and their desired values:

```
param c(1) 153 c(2) .68 c(3) .15
```

sets C(1) = 153, C(2) = .68, and C(3) = .15. All of the other elements of the coefficient vector are left unchanged.

Lastly, if you want to use estimated coefficients from another equation, select **Proc/Update Coefs from Equation** from the equation window toolbar.

For nonlinear least squares problems or situations where you specify the starting values, bear in mind that:

- The objective function must be defined at the starting values. For example, if your objective function contains the expression 1/C(1), then you cannot set C(1) to zero. Similarly, if the objective function contains LOG(C(2)), then C(2) must be greater than zero.

- A poor choice of starting values may cause the nonlinear least squares algorithm to fail. EViews begins nonlinear estimation by taking derivatives of the objective function with respect to the parameters, evaluated at these values. If these derivatives are not well behaved, the algorithm may be unable to proceed.

  If, for example, the starting values are such that the derivatives are all zero, you will immediately see an error message indicating that EViews has encountered a "Near Singular Matrix", and the estimation procedure will stop.

- Unless the objective function is globally concave, iterative algorithms may stop at a local optimum. There will generally be no evidence of this fact in any of the output from estimation.

  If you are concerned with the possibility of local optima, you may wish to select various starting values and see whether the estimates converge to the same values. One common suggestion is to estimate the model and then randomly alter each of the estimated coefficients by some percentage, then use these new coefficients as starting values in estimation.

Iteration and Convergence Options

There are two common iteration stopping rules: based on the change in the objective function, or based on the change in parameters. The convergence rule used in EViews is based upon changes in the parameter values. This rule is generally conservative, since the change in the objective function may be quite small as we approach the optimum (this is how we choose the direction), while the parameters may still be changing.

The exact rule in EViews is based on comparing the norm of the change in the parameters with the norm of the current parameter values. More specifically, the convergence test is:

$$\frac{\left\|\theta_{(i+1)} - \theta_{(i)}\right\|_2}{\left\|\theta_{(i)}\right\|_2} \le tol \tag{C.1}$$

where θ is the vector of parameters, $\|x\|_2$ is the 2-norm of x, and *tol* is the specified tolerance. However, before taking the norms, each parameter is scaled based on the largest observed norm across iterations of the derivative of the least squares residuals with respect to that parameter. This automatic scaling system makes the convergence criteria more robust to changes in the scale of the data, but does mean that restarting the optimization from the final converged values may cause additional iterations to take place, due to slight changes in the automatic scaling value when started from the new parameter values.

The estimation process achieves convergence if the stopping rule is reached using the tolerance specified in the **Convergence** edit box of the Estimation Dialog or the Estimation Options Dialog. By default, the box will be filled with the tolerance value specified in the global estimation options, or if the estimation object has previously been estimated, it will be filled with the convergence value specified for the last set of estimates.

EViews may stop iterating even when convergence is not achieved. This can happen for two reasons. First, the number of iterations may have reached the prespecified upper bound. In this case, you should reset the maximum number of iterations to a larger number and try iterating until convergence is achieved.

Second, EViews may issue an error message indicating a "Failure to improve"after a number of iterations. This means that even though the parameters continue to change, EViews could not find a direction or step size that improves the objective function. This can happen when the objective function is ill-behaved; you should make certain that your model is identified. You might also try other starting values to see if you can approach the optimum from other directions.

Lastly, EViews may converge, but warn you that there is a singularity and that the coefficients are not unique. In this case, EViews will not report standard errors or *t*-statistics for the coefficient estimates.

Derivative Computation Options

In many EViews estimation procedures, you can specify the form of the function for the mean equation. For example, when estimating a regression model, you may specify an arbitrary nonlinear expression in the coefficients. In these cases, when estimating the model, EViews will compute derivatives of the user-specified function.

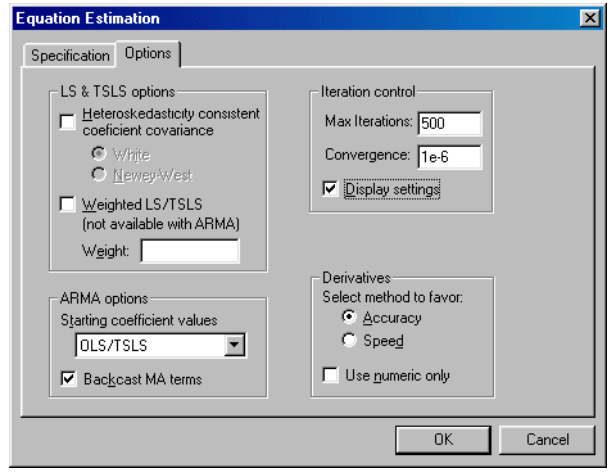
EViews uses two techniques for evaluating derivatives: numeric (finite difference) and analytic. The approach that is used depends upon the nature of the optimization problem and any user-defined settings:

- In most cases, EViews offers the user the choice of computing either analytic or numeric derivatives. By default, EViews will fill the options dialog with the global estimation settings. If the **Use numeric only** setting is chosen, EViews will only compute the derivatives using finite difference methods. If this setting is not checked, EViews will attempt to compute analytic derivatives, and will use numeric derivatives only where necessary.
- EViews will ignore the numeric derivative setting and use an analytic derivative whenever a coefficient derivative is a constant value.
- For some procedures where the range of specifications allowed is limited, EViews always uses analytic first and/or second derivatives. VARs, pools, binary models

(probit, logit, etc.), count models, censored (tobit) models, and ordered models all fall into this category.

- The derivatives with respect to the AR coefficients in an ARMA specification are always computed analytically while those with respect to the MA coefficients are computed numerically.
- In a limited number of cases, EViews will always use numeric derivatives. For the moment, GARCH and state space models always use numeric derivatives. As noted above, MA coefficient derivatives are always computed numerically.
- Logl objects always use numeric derivatives unless you provide the analytic derivatives in the specification.

Where relevant, the estimation options dialog allows you to control the method of taking derivatives. For example, the options dialog for standard regression allows you to override the use of EViews analytic derivatives, and to choose between favoring speed or accuracy in the computation of any numeric derivatives (note that the additional LS and TSLS options are discussed in detail in Chapter 16, "Additional Regression Methods", beginning on page 461).



Computing the more accurate numeric derivatives requires additional objective function evaluations. While the algorithms may change in future versions, at present, EViews computes numeric derivatives using either a one-sided finite difference (favor speed), or using a four-point routine using Richardson extrapolation (favor precision). Additional details are provided in Kincaid and Cheney (1996).

Analytic derivatives will often be faster and more accurate than numeric derivatives, especially if the analytic derivatives have been simplified and carefully optimized to remove common subexpressions. Numeric derivatives will sometimes involve fewer floating point operations than analytic, and in these circumstances, may be faster.

Optimization Algorithms

Given the importance of the proper setting of EViews estimation options, it may prove useful to review briefly various basic optimization algorithms used in nonlinear estimation. Recall that the problem faced in non-linear estimation is to find the values of parameters θ that optimize (maximize or minimize) an objective function $F(\theta)$.

Iterative optimization algorithms work by taking an initial set of values for the parameters, say $\theta_{(0)}$, then performing calculations based on these values to obtain a better set of parameter values, $\theta_{(1)}$. This process is repeated for $\theta_{(2)}$, $\theta_{(3)}$ and so on until the objective function F no longer improves between iterations.

There are three main parts to the optimization process: (1) obtaining the initial parameter values, (2) updating the candidate parameter vector θ at each iteration, and (3) determining when we have reached the optimum.

If the objective function is globally concave so that there is a single maximum, any algorithm which improves the parameter vector at each iteration will eventually find this maximum (assuming that the size of the steps taken does not become negligible). If the objective function is not globally concave, different algorithms may find different local maxima, but all iterative algorithms will suffer from the same problem of being unable to tell apart a local and a global maximum.

The main thing that distinguishes different algorithms is how quickly they find the maximum. Unfortunately, there are no hard and fast rules. For some problems, one method may be faster, for other problems it may not. EViews provides different algorithms, and will often let you choose which method you would like to use.

The following sections outline these methods. The algorithms used in EViews may be broadly classified into three types: *second derivative* methods, *first derivative* methods, and *derivative free* methods. EViews' second derivative methods evaluate current parameter values and the first and second derivatives of the objective function for every observation. First derivative methods use only the first derivatives of the objective function during the iteration process. As the name suggests, derivative free methods do not compute derivatives.

Second Derivative Methods

For binary, ordered, censored, and count models, EViews can estimate the model using Newton-Raphson or *quadratic hill-climbing*.

Newton-Raphson

Candidate values for the parameters $\theta_{(1)}$ may be obtained using the method of Newton-Raphson by linearizing the first order conditions $\partial F/\partial\theta$ at the current parameter values, $\theta_{(i)}$:

$$\begin{aligned} g_{(i)} + H_{(i)}(\theta_{(i+1)} - \theta_{(i)}) &= 0 \\ \theta_{(i+1)} &= \theta_{(i)} - H_{(i)}^{-1} g_{(i)} \end{aligned} \tag{C.2}$$

where g is the gradient vector $\partial F/\partial\theta$, and H is the Hessian matrix $\partial^2 F/\partial\theta^2$.

If the function is quadratic, Newton-Raphson will find the maximum in a single iteration. If the function is not quadratic, the success of the algorithm will depend on how well a local quadratic approximation captures the shape of the function.

Quadratic hill-climbing (Goldfeld-Quandt)

This method, which is a straightforward variation on Newton-Raphson, is sometimes attributed to Goldfeld and Quandt. Quadratic hill-climbing modifies the Newton-Raphson algorithm by adding a correction matrix (or ridge factor) to the Hessian. The quadratic hill-climbing updating algorithm is given by:

$$\theta_{(i+1)} = \theta_{(i)} - \tilde{H}_{(i)}^{-1} g_{(i)} \qquad \text{where } -\tilde{H}_{(i)} = -H_{(i)} + \alpha I \tag{C.3}$$

where I is the identity matrix and α is a positive number that is chosen by the algorithm.

The effect of this modification is to push the parameter estimates in the direction of the gradient vector. The idea is that when we are far from the maximum, the local quadratic approximation to the function may be a poor guide to its overall shape, so we may be better off simply following the gradient. The correction may provide better performance at locations far from the optimum, and allows for computation of the direction vector in cases where the Hessian is near singular.

For models which may be estimated using second derivative methods, EViews uses quadratic hill-climbing as its default method. You may elect to use traditional Newton-Raphson, or the first derivative methods described below, by selecting the desired algorithm in the Options menu.

Note that asymptotic standard errors are always computed from the unmodified Hessian once convergence is achieved.

First Derivative Methods

Second derivative methods may be computationally costly since we need to evaluate the $k(k+1)/2$ elements of the second derivative matrix at every iteration. Moreover, second derivatives calculated may be difficult to compute accurately. An alternative is to employ

methods which require only the first derivatives of the objective function at the parameter values.

For general nonlinear models (nonlinear least squares, ARCH and GARCH, nonlinear system estimators, GMM, State Space), EViews provides two first derivative methods: Gauss-Newton/BHHH or Marquardt.

Gauss-Newton/BHHH

This algorithm follows Newton-Raphson, but replaces the negative of the Hessian by an approximation formed from the sum of the outer product of the gradient vectors for each observation's contribution to the objective function. For least squares and log likelihood functions, this approximation is asymptotically equivalent to the actual Hessian when evaluated at the parameter values which maximize the function. When evaluated away from the maximum, this approximation may be quite poor.

The algorithm is referred to as *Gauss-Newton* for general nonlinear least squares problems, and often attributed to Berndt, Hall, Hall, and Hausman (*BHHH*) for maximum likelihood problems.

The advantages of approximating the negative Hessian by the outer product of the gradient are that (1) we need to evaluate only the first derivatives, and (2) the outer product is necessarily positive semi-definite. The disadvantage is that, away from the maximum, this approximation may provide a poor guide to the overall shape of the function, so that more iterations may be needed for convergence.

Marquardt

The Marquardt algorithm modifies the Gauss-Newton algorithm in exactly the same manner as quadratic hill climbing modifies the Newton-Raphson method (by adding a correction matrix (or ridge factor) to the Hessian approximation).

The ridge correction handles numerical problems when the outer product is near singular and may improve the convergence rate. As above, the algorithm pushes the updated parameter values in the direction of the gradient.

For models which may be estimated using first derivative methods, EViews uses Marquardt as its default method. You may elect to use traditional Gauss-Newton via the Options menu.

Note that asymptotic standard errors are always computed from the unmodified (Gauss-Newton) Hessian approximation once convergence is achieved.

Choosing the step size

At each iteration, we can search along the given direction for the optimal step size. EViews performs a simple trial-and-error search at each iteration to determine a step size λ that improves the objective function. This procedure is sometimes referred to as *squeezing* or *stretching*.

Note that while EViews will make a crude attempt to find a good step, λ is not actually optimized at each iteration since the computation of the direction vector is often more important than the choice of the step size. It is possible, however, that EViews will be unable to find a step size that improves the objective function. In this case, EViews will issue an error message.

EViews also performs a crude trial-and-error search to determine the scale factor α for Marquardt and quadratic hill-climbing methods.

Derivative free methods

Other optimization routines do not require the computation of derivatives. The *grid search* is a leading example. Grid search simply computes the objective function on a grid of parameter values and chooses the parameters with the highest values. Grid search is computationally costly, especially for multi-parameter models.

EViews uses (a version of) grid search for the exponential smoothing routine.

Nonlinear Equation Solution Methods

When solving a nonlinear equation system, EViews first analyzes the system to determine if the system can be separated into two or more blocks of equations which can be solved sequentially rather than simultaneously. Technically, this is done by using a graph representation of the equation system where each variable is a vertex and each equation provides a set of edges. A well known algorithm from graph theory is then used to find the strongly connected components of the directed graph.

Once the blocks have been determined, each block is solved for in turn. If the block contains no simultaneity, each equation in the block is simply evaluated once to obtain values for each of the variables.

If a block contains simultaneity, the equations in that block are solved by either a Gauss-Seidel or Newton method, depending on how the solver options have been set.

Gauss-Seidel

By default, EViews uses the Gauss-Seidel method when solving systems of nonlinear equations. Suppose the system of equations is given by:

$$\begin{aligned} x_1 &= f_1(x_1, x_2, \ldots, x_N, z) \\ x_2 &= f_2(x_1, x_2, \ldots, x_N, z) \\ &\vdots \\ x_N &= f_N(x_1, x_2, \ldots, x_N, z) \end{aligned} \tag{C.4}$$

where x are the endogenous variables and z are the exogenous variables.

The problem is to find a fixed point such that $x = f(x, z)$. Gauss-Seidel employs an iterative updating rule of the form:

$$x^{(i+1)} = f(x^{(i)}, z). \tag{C.5}$$

to find the solution. At each iteration, EViews solves the equations in the order that they appear in the model. If an endogenous variable that has already been solved for in that iteration appears later in some other equation, EViews uses the value as solved in that iteration. For example, the k-th variable in the i-th iteration is solved by :

$$x_k^{(i)} = f_k(x_1^{(i)}, x_2^{(i)}, \ldots, x_{k-1}^{(i)}, x_k^{(i-1)}, x_{k+1}^{(i-1)}, \ldots, x_N^{(i-1)}, z). \tag{C.6}$$

The performance of the Gauss-Seidel method can be affected be reordering of the equations. If the Gauss-Seidel method converges slowly or fails to converge, you should try moving the equations with relatively few and unimportant right-hand side endogenous variables so that they appear early in the model.

Newton's Method

Newton's method for solving a system of nonlinear equations consists of repeatedly solving a local linear approximation to the system.

Consider the system of equations written in implicit form:

$$F(x, z) = 0 \tag{C.7}$$

where F is the set of equations, x is the vector of endogenous variables and z is the vector of exogenous variables.

In Newton's method, we take a linear approximation to the system around some values x^* and z^* :

$$F(x, z) = F(x^*, z^*) + \frac{\partial}{\partial x} F(x^*, z^*)\Delta x = 0 \tag{C.8}$$

and then use this approximation to construct an iterative procedure for updating our current guess for x :

$$x_{t+1} = x_t - \left[\frac{\partial}{\partial x} F(x_t, z^*)\right]^{-1} F(x_t, z^*) \tag{C.9}$$

where raising to the power of -1 denotes matrix inversion.

The procedure is repeated until the changes in x_t between periods are smaller than a specified tolerance.

Note that in contrast to Gauss-Seidel, the ordering of equations under Newton does not affect the rate of convergence of the algorithm.

Appendix D. Gradients and Derivatives

Many EViews estimation objects provide built-in routines for examining the gradients and derivatives of your specifications. You can, for example, use these tools to examine the analytic derivatives of your nonlinear regression specification in numeric or graphical form, or you can save the gradients from your estimation routine for specification tests.

The gradient and derivative views may be accessed from most estimation objects by selecting **View/Gradients and Derivatives** or, in some cases, **View/Gradients**, and then selecting the appropriate view.

If you wish to save the numeric values of your gradients and derivatives, you will need to use the gradient and derivative procedures. These procs may be accessed from the main **Proc** menu.

Note that all views and procs are not available for every estimation object or every estimation technique.

Gradients

EViews provides you with the ability to examine and work with the gradients of the objective function for a variety of estimation objects. Examining these gradients can provide useful information for evaluating the behavior of your nonlinear estimation routine, or can be used as the basis of various tests of specification.

Since EViews provides a variety of estimation methods and techniques, the notion of a gradient is a bit difficult to describe in casual terms. EViews will generally report the values of the first-order conditions used in estimation. To take the simplest example, ordinary least squares minimizes the sum-of-squared residuals:

$$S(\beta) = \sum_t (y_t - X_t'\beta)^2 \tag{D.1}$$

The first-order conditions for this objective function are obtained by differentiating with respect to β, yielding

$$\sum_t -2(y_t - X_t'\beta)X_t \tag{D.2}$$

EViews allows you to examine both the sum and the corresponding average, as well as the value for each of the individual observations. Furthermore, you can save the individual values in series for subsequent analysis.

The individual gradient computations are summarized in the following table:

Least squares	$g_t = -2(y_t - f_t(X_t, \beta))\left(\frac{\partial f_t(X_t, \beta)}{\partial \beta}\right)$
Weighted least squares	$g_t = -2(y_t - f_t(X_t, \beta))w_t^2\left(\frac{\partial f_t(X_t, \beta)}{\partial \beta}\right)$
Two-stage least squares	$g_t = -2(y_t - f_t(X_t, \beta))P_t\left(\frac{\partial f_t(X_t, \beta)}{\partial \beta}\right)$
Weighted two-stage least squares	$g_t = -2(y_t - f_t(X_t, \beta))w_t\tilde{P}_t w_t\left(\frac{\partial f_t(X_t, \beta)}{\partial \beta}\right)$
Maximum likelihood	$g_t = \frac{\partial l_t(X_t, \beta)}{\partial \beta}$

where P and $\tilde{P}$ are the projection matrices corresponding to the expressions for the estimators in Chapter 16, "Additional Regression Methods", beginning on page 461, and where l is the log likelihood contribution function.

Note that the expressions for the regression gradients are adjusted accordingly in the presence of ARMA error terms.

Gradient Summary

To view the summary of the gradients, select **View/Gradients and Derivatives/Gradient Summary**, or **View/Gradients/Summary**. EViews will display a summary table showing the sum, mean, and Newton direction associated with the gradients. Here is an example table from a nonlinear least squares estimation equation:

Gradients of the objective function at estimated parameters
Equation: EQ1
Method: Least Squares
Specification: Y = C(1)*EXP(-C(2)*X) + C(3)*EXP(-((X -C(4))^2) / C(5)^2) + C(6)*EXP(-((X-C(7))^2) / C(8)^2)
Computed using analytic derivatives

Coefficient	Sum	Mean	Newton Dir.
C(1)	-3.49E-09	-1.40E-11	-2.43E-12
C(2)	-2.72E-06	-1.09E-08	-7.74E-16
C(3)	-7.76E-09	-3.11E-11	-9.93E-12
C(4)	3.85E-09	1.54E-11	1.04E-14
C(5)	8.21E-09	3.29E-11	1.97E-13
C(6)	1.21E-09	4.84E-12	-2.20E-12
C(7)	-9.16E-10	-3.67E-12	3.53E-14
C(8)	2.85E-08	1.14E-10	3.95E-13

There are several things to note about this table. The first line of the table indicates that the gradients have been computed at estimated parameters. If you ask for a gradient view for an estimation object that has not been successfully estimated, EViews will compute the gradients at the current parameter values and will note this in the table. This behavior allows you to diagnose unsuccessful estimation problems using the gradient values.

Second, you will note that EViews informs you that the gradients were computed using analytic derivatives. EViews will also inform you if the specification is linear, if the derivatives were computed numerically, or if EViews used a mixture of analytic and numeric techniques. We remind you that all MA coefficient derivatives are computed numerically.

Lastly, there is a table showing the sum and mean of the gradients as well as a column labeled "Newton Dir.". The column reports the non-Marquardt adjusted Newton direction used in first-derivative iterative estimation procedures (see "First Derivative Methods" on page 957).

In the example above, all of the values are "close" to zero. While one might expect these values always to be close to zero when evaluated at the estimated parameters, there are a number of reasons why this will not always be the case. First, note that the sum and mean values are highly scale variant so that changes in the scale of the dependent and independent variables may lead to marked changes in these values. Second, you should bear in mind that while the Newton direction is *related* to the terms used in the optimization procedures, EViews' test for convergence does not directly use the Newton direction. Third, some of the iteration options for system estimation do not iterate coefficients or weights fully to convergence. Lastly, you should note that the values of these gradients are sensitive to the accuracy of any numeric differentiation.

Gradient Table and Graph

There are a number of situations in which you may wish to examine the individual contributions to the gradient vector. For example, one common source of singularity in nonlinear estimation is the presence of very small combined with very large gradients at a given set of coefficient values.

EViews allows you to examine your gradients in two ways: as a spreadsheet of values, or as line graphs, with each set of coefficient gradients plotted in a separate graph. Using these tools, you can examine your data for observations with outlier values for the gradients.

Gradient Series

You can save the individual gradient values in series using the **Make Gradient Group** procedure. EViews will create a new group containing series with names of the form GRAD## where ## is the next available name.

Note that when you store the gradients, EViews will fill the series for the full workfile range. If you view the series, make sure to set the workfile sample to the sample used in estimation if you want to reproduce the table displayed in the gradient views.

Application to LM Tests

The gradient series are perhaps most useful for carrying out Lagrange multiplier tests for nonlinear models by running what is known as artificial regressions (Davidson and MacKinnon 1993, Chapter 6). A generic artificial regression for hypothesis testing takes the form of regressing:

$$\tilde{u}_t \text{ on } \left(\frac{\partial f_t(X_t, \tilde{\beta})}{\partial \beta}\right) \text{ and } Z_t \tag{D.3}$$

where $\tilde{u}$ are the estimated residuals under the restricted (null) model, and $\tilde{\beta}$ are the estimated coefficients. The Z are a set of "misspecification indicators" which correspond to departures from the null hypothesis.

An example program ("GALLANT2.PRG") for performing an LM auxiliary regression test is provided in your EViews installation directory.

Gradient Availability

The gradient views are currently available for the equation, logl, sspace and system objects. The views are not, however, currently available for equations estimated by GMM or ARMA equations specified by expression.

Derivatives

EViews employs a variety of rules for computing the derivatives used by iterative estimation procedures. These rules, and the user-defined settings that control derivative taking, are described in detail in "Derivative Computation Options" on page 954.

In addition, EViews provides both object views and object procedures which allow you to examine the effects of those choices, and the results of derivative taking. These views and procedures provide you with quick and easy access to derivatives of your user-specified functions.

It is worth noting that these views and procedures are not available for all estimation techniques. For example, the derivative views are currently not available for binary models since only a limited set of specifications are allowed.

Derivative Description

The **Derivative Description** view provides a quick summary of the derivatives used in estimation.

For example, consider the simple nonlinear regression model:

$$y_t = c(1)(1 - \exp(-c(2)x_t)) + \epsilon_t \tag{D.4}$$

Following estimation of this single equation, we can display the description view by selecting **View/Gradients and Derivatives.../Derivative Description.**

Derivatives of the equation specification
Equation: EQ1
Method: Least Squares
Specification: RESID = Y - (C(1)*(1 - EXP(-C(2)*X)))
Computed using analytic derivatives

Coefficient	Derivative of Specification
C(1)	-1 + exp(-c(2) * x)
C(2)	-c(1) * x * exp(-c(2) * x)

There are three parts to the output from this view. First, the line labeled "Specification:" describes the equation specification that we are estimating. You will note that we have written the specification in terms of the implied residual from our specification.

The next line describes the method used to compute the derivatives used in estimation. Here, EViews reports that the derivatives were computed analytically.

Lastly, the bottom portion of the table displays the expressions for the derivatives of the regression function with respect to each coefficient. Note that the derivatives are in terms of the implied residual so that the signs of the expressions have been adjusted accordingly.

In this example, all of the derivatives were computed analytically. In some cases, however, EViews will not know how to take analytic derivatives of your expression with respect to one or more of the coefficient. In this situation, EViews will use analytic expressions where possible, and numeric where necessary, and will report which type of derivative was used for each coefficient.

Suppose, for example, that we estimate:

$$y_t = c(1)(1 - \exp(-\phi(c(2)x_t))) + \epsilon_t \tag{D.5}$$

where ϕ is the standard normal density function. The derivative view of this equation is

Derivatives of the equation specification
Equation: EQ1
Method: Least Squares
Specification: RESID = Y - (C(1)*(1 - EXP(-@DNORM(C(2)*X))))
Computed using analytic derivatives
Use accurate numeric derivatives where necessary

Coefficient	Derivative of Specification
C(1)	-1 + exp(-@dnorm(c(2) * x))
C(2)	--- accurate numeric ---

Here, EViews reports that it attempted to use analytic derivatives, but that it was forced to use a numeric derivative for C(2) (since it has not yet been taught the derivative of the @DNORM function).

If we set the estimation option so that we only compute fast numeric derivatives, the view would change to

Derivatives of the equation specification
Equation: EQ1
Method: Least Squares
Specification: RESID = Y - (C(1)*(1 - EXP(-C(2)*X)))
Computed using fast numeric derivatives

Coefficient	Derivative of Specification
C(1)	--- fast numeric ---
C(2)	--- fast numeric ---

to reflect the different method of taking derivatives.

If your specification contains autoregressive terms, EViews will only compute the derivatives with respect to the regression part of the equation. The presence of the AR components is, however, noted in the description view.

Derivatives of the equation specification
Equation: EQ1
Method: Least Squares
Specification: [AR(1)=C(3)] = Y - (C(1)*(1 - EXP(-C(2)*X)))
Computed using analytic derivatives

Coefficient	Derivative of Specification*
C(1)	-1 + exp(-c(2) * x)
C(2)	-c(1) * x * exp(-c(2) * x)

*Note: derivative expressions do not account for AR components

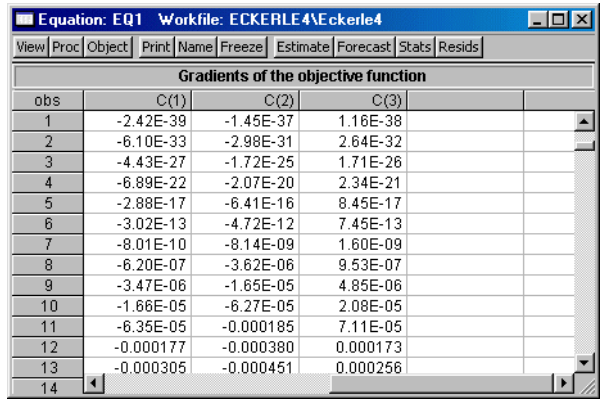
Recall that the derivatives of the objective function with respect to the AR components are always computed analytically using the derivatives of the regression specification, and the lags of these values.

One word of caution about derivative expressions. For many equation specifications, analytic derivative expressions will be quite long. In some cases, the analytic derivatives will be longer than the space allotted to them in the table output. You will be able to identify these cases by the trailing "..." in the expression.

To see the entire expression, you will have to create a table object and then resize the appropriate column. Simply click on the **Freeze** button on the toolbar to create a table object, and then highlight the column of interest. Click on **Width** on the table toolbar and enter in a larger number.

Derivative Table and Graph

Once we obtain estimates of the parameters of our nonlinear regression model, we can examine the values of the derivatives at the estimated parameter values. Simply select **View/Gradients and Derivatives...** to see a spreadsheet view or line graph of the values of the derivatives for each coefficient:


Equation: EQ1 Workfile: ECKERLE4\Eckerle4
View Proc Object Print Name Freeze Estimate Forecast Stats Resids
Gradients of the objective function

obs	C(1)	C(2)	C(3)
1	-2.42E-39	-1.45E-37	1.16E-38
2	-6.10E-33	-2.98E-31	2.64E-32
3	-4.43E-27	-1.72E-25	1.71E-26
4	-6.89E-22	-2.07E-20	2.34E-21
5	-2.88E-17	-6.41E-16	8.45E-17
6	-3.02E-13	-4.72E-12	7.45E-13
7	-8.01E-10	-8.14E-09	1.60E-09
8	-6.20E-07	-3.62E-06	9.53E-07
9	-3.47E-06	-1.65E-05	4.85E-06
10	-1.66E-05	-6.27E-05	2.08E-05
11	-6.35E-05	-0.000185	7.11E-05
12	-0.000177	-0.000380	0.000173
13	-0.000305	-0.000451	0.000256
14			

This spreadsheet view displays the value of the derivatives for each observation in the standard spreadsheet form. The graph view, plots the value of each of these derivatives for each coefficient.

Derivative Series

You can save the derivative values in series for later use. Simply select **Proc/Make Derivative Group** and EViews will create an untitled group object containing the new series. The series will be named DERIV##, where ## is a number associated with the next available free name. For example, if you have the objects DERIV01 and DERIV02, but not DERIV03 in the workfile, EViews will save the next derivative in the series DERIV03.

Appendix E. Information Criteria

As part of the output for most regression procedures, EViews reports various information criteria. The information criteria are often used as a guide in model selection (see for example, Grasa 1989).

The Kullback-Leibler quantity of information contained in a model is the distance from the "true" model and is measured by the log likelihood function. The notion of an information criterion is to provide a measure of information that strikes a balance between this measure of goodness of fit and parsimonious specification of the model. The various information criteria differ in how to strike this balance.

Definitions

The basic information criteria are given by:

Akaike info criterion (AIC)	$-2(l/T)+2(k/T)$
Schwarz criterion (SC)	$-2(l/T)+k\log(T)/T$
Hannan-Quinn criterion (HQ)	$-2(l/T)+2k\log(\log(T))/T$

Let l be the value of the log of the likelihood function with the k parameters estimated using T observations. The various information criteria are all based on –2 times the average log likelihood function, adjusted by a penalty function.

In addition to the information criteria described above, there are specialized information criteria that are used in by EViews when computing unit root tests:

Modified AIC (MAIC)	$-2(l/T)+2((k+\tau)/T)$
Modified SIC (MSIC)	$-2(l/T)+(k+\tau)\log(T)/T$
Modified Hannan-Quinn (MHQ)	$-2(l/T)+2(k+\tau)\log(\log(T))/T$

where the modification factor τ is computed as:

$$\tau = \alpha^2 \sum_t \tilde{y}_{t-1}^2 / \sigma^2 \tag{E.1}$$

for $\tilde{y}_t \equiv y_t$ when computing the ADF test equation (17.43), and for $\tilde{y}_t$ as defined in ("Autoregressive Spectral Density Estimator" beginning on page 528) when estimating the frequency zero spectrum (see Ng and Perron, 2002, for a discussion of the modified information criteria).

Note also that:

- The Hannan-Quinn criterion is reported only for binary, ordered, censored, and count models.
- The definitions used by EViews may differ slightly from those used by some authors. For example, Grasa (1989, equation 3.21) does not divide the AIC by n. Other authors omit inessential constants of the Gaussian log likelihood (generally, the terms involving 2π).

  While very early versions of EViews reported information criteria that omitted inessential constant terms, the current version of EViews always uses the value of the full likelihood function. All of your equation objects estimated in earlier versions of EViews will automatically be updated to reflect this change. You should, however, keep this fact in mind when comparing results from frozen table objects or printed output from previous versions.
- For systems of equations, where applicable, the information criteria are computed using the full system log likelihood. The log likelihood value is computed assuming a multivariate normal (Gaussian) distribution as:

$$l = -\frac{TM}{2}(1 + \log 2\pi) - \frac{T}{2}\log\left|\hat{\Omega}\right| \tag{E.2}$$

  where

$$\left|\hat{\Omega}\right| = \det(\sum_i \hat{\epsilon}\hat{\epsilon}'/T) \tag{E.3}$$

  M is the number of equations. Note that these expressions are only strictly valid when you there are equal numbers of observations for each equation. When your system is unbalanced, EViews replaces these expressions with the appropriate summations.

Using Information Criteria as a Guide to Model Selection

As a user of these information criteria as a model selection guide, you select the model with the smallest information criterion.

The information criterion has been widely used in time series analysis to determine the appropriate length of the distributed lag. Lütkepohl (1991, Chapter 4) presents a number of results regarding consistent lag order selection in VAR models.

You should note, however, that the criteria depend on the unit of measurement of the dependent variable y. For example, you cannot use information criteria to select between a model with dependent variable y and one with $\log(y)$.

References

Abramowitz, M. and I. A. Stegun (1964). *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*, New York: Dover Publications.

Aitchison, J. and S.D. Silvey (1957). "The Generalization of Probit Analysis to the Case of Multiple Responses," *Biometrika*, 44, 131–140.

Agresti, Alan (1996). *An Introduction to Categorical Data Analysis*, New York: John Wiley & Sons.

Amemiya, Takeshi (1983). "Nonlinear Regression Models," Chapter 6 in Z. Griliches and M. D. Intriligator (eds.), *Handbook of Econometrics, Volume 1*, Amsterdam: Elsevier Science Publishers B.V.

Amisano, Gianni and Carlo Giannini (1997). *Topics in Structural VAR Econometrics*, 2nd ed, Berlin: Springer-Verlag.

Anderson, T. W. and C. Hsiao (1981). "Estimation of Dynamic Models with Error Components," *Journal of the American Statistical Association*, 76, 598-606.

Anderson, T. W. and D. A. Darling (1952). "Asymptotic Theory of Certain Goodness of Fit Criteria Based on Stochastic Processes," *Annals of Mathematical Statistics*, 23, 193-212.

Anderson, T. W. and D. A. Darling (1954), "A Test of Goodness of Fit," *Journal of the American Statistical Association*, 49, 765-769.

Andrews, Donald W. K. (1988a). "Chi-Square Diagnostic Tests for Econometric Models: Theory," *Econometrica*, 56, 1419–1453.

Andrews, Donald W. K. (1988b). "Chi-Square Diagnostic Tests for Econometric Models: Introduction and Applications," *Journal of Econometrics*, 37, 135–156.

Andrews, Donald W. K. (1991). "Heteroskedasticity and Autocorrelation Consistent Covariance Matrix Estimation," *Econometrica*, 59, 817–858.

Andrews, Donald W. K. and J. Christopher Monahan (1992). "An Improved Heteroskedasticity and Autocorrelation Consistent Covariance Matrix Estimator," *Econometrica*, 60, 953–966.

Arellano, M. (1987). "Computing Robust Standard Errors for Within-groups Estimators," *Oxford Bulletin of Economics and Statistics*, 49, 431-434.

Arellano, M., and Bond, S. R. (1991). "Some Tests of Specification for Panel Data: Monte Carlo Evidence and an Application to Employment Equations," *Review of Economic Studies*, 58, 277–297.

Arellano, M., and Bover, O. (1995). "Another Look at the Instrumental Variables Estimation of Error-components Models," *Journal of Econometrics*, 68, 29–51.

Baltagi, Badi H. and Young-Jae Chang (1994). "Incomplete Panels: A Comparative Study of Alternative Estimators for the Unbalanced One-way Error Component Regression Model," *Journal of Econometrics*, 62, 67-89.

Baltagi, Badi H., Seuck H. Song, and Byoung C. Jung (2002). "A Comparative Study Of Alternative Estimators for the Unbalanced Two-way Error Component Regression Model," *Econometric Journal*, 5, 480-493.

Baltagi, Badi H. (2001). *Econometric Analysis of Panel Data, Second Edition*, West Sussex, England: John Wiley & Sons.

Baxter, Marianne and Robert G. King (1999). "Measuring Business Cycles: Approximate Band-Pass Filters For Economic Time Series," *Review of Economics and Statistics*, 81, 575–593.

Beck, Nathaniel and Jonathan N. Katz (1995). "What to Do (and Not to Do) With Time-series Cross-section Data," *American Political Science Review*, 89(3), 634-647.

Bergmann, Reinhard, John Ludbrook, and Will P. J. M. Spooren (2000). "Different Outcomes of the Wilcoxon-Mann-Whitney Test From Different Statistical Packages," *The American Statistician*, 45(1), 72–77.

Berndt, Ernst R. and David O. Wood (1975). "Technology, Prices and the Derived Demand for Energy," *Review of Economics and Statistics*, 57(3), 259-268.

Bhargava, A. (1986). "On the Theory of Testing for Unit Roots in Observed Time Series," *Review of Economic Studies*, 53, 369-384.

Blanchard, Olivier (1989). "A Traditional Interpretation of Macroeconomic Fluctuations," *American Economic Review*, 79, 1146-1164.

Blanchard, Olivier and Danny Quah (1989). "The Dynamic Effects of Aggregate Demand and Aggregate Supply Disturbances," *American Economic Review*, 79, 655-673.

Bollerslev, Tim (1986). "Generalized Autoregressive Conditional Heteroskedasticity," *Journal of Econometrics*, 31, 307–327.

Bollerslev, Tim, Ray Y. Chou, and Kenneth F. Kroner (1992). "ARCH Modeling in Finance: A Review of the Theory and Empirical Evidence," *Journal of Econometrics,* 52, 5–59.

Bollerslev, Tim and Jeffrey M. Wooldridge (1992). "Quasi-Maximum Likelihood Estimation and Inference in Dynamic Models with Time Varying Covariances," *Econometric Reviews,* 11, 143–172.

Bollerslev, Tim, Robert F. Engle and Daniel B. Nelson (1994). "ARCH Models," Chapter 49 in Robert F. Engle and Daniel L. McFadden (eds.), *Handbook of Econometrics, Volume 4*, Amsterdam: Elsevier Science B.V.

Boswijk, H. Peter (1995). "Identifiability of Cointegrated Systems," Technical Report, Tinbergen Institute.

Bowerman, Bruce L. and Richard T. O'Connell (1979). *Time Series and Forecasting: An Applied Approach,* New York: Duxbury Press.

Box, George E. P. and Gwilym M. Jenkins (1976). *Time Series Analysis: Forecasting and Control*, Revised Edition, Oakland, CA: Holden-Day.

Box, George E. P. and D. A. Pierce (1970). "Distribution of Residual Autocorrelations in Autoregressive Integrated Moving Average Time Series Models," *Journal of the American Statistical Association*, 65, 1509–1526.

Breitung, Jörg (2000). "The Local Power of Some Unit Root Tests for Panel Data," in B. Baltagi (ed.), *Advances in Econometrics, Vol. 15: Nonstationary Panels, Panel Cointegration, and Dynamic Panels*, Amsterdam: JAI Press, p. 161–178.

Brock, William, Davis Dechert, Jose Sheinkman & Blake LeBaron (1996). "A Test for Independence Based on the Correlation Dimension," *Econometric Reviews*, August, 15(3), 197–235.

Brown, R. L., J. Durbin, and J. M. Evans (1975). "Techniques for Testing the Constancy of Regression Relationships Over Time," *Journal of the Royal Statistical Society, Series B*, 37, 149–192.

Brown, M. B. and A. B. Forsythe (1974a). "Robust Tests for the Equality of Variances," *Journal of the American Statistical Association*, 69, 364–367.

Brown, M. B. and A. B. Forsythe (1974b). "The Small Sample Behavior of Some Test Statistics which Test the Equality of Several Means," *Technometrics*, 16, 129–132.

Cameron, A. Colin and Pravin K. Trivedi (1990). "Regression-based Tests for Overdispersion in the Poisson Model," *Journal of Econometrics*, 46, 347–364.

Campbell, John Y. and Pierre Perron (1991). "Pitfalls and Opportunities: What Macroeconomists Should Know about Unit Roots," *NBER Macroeconomics Annual*, 141–201.

Chambers, John M., William S. Cleveland, Beat Kleiner, Paul A. Tukey (1983). *Graphical Methods for Data Analysis*, Murray Hill, NJ: Wadsworth & Brooks/Cole Publishing Company.

Chesher, A., T. Lancaster, and M. Irish (1985). "On Detecting the Failure of Distributional Assumptions," *Annales de L'Insee*, 59/60, 7–44.

Chesher, A. and M. Irish (1987). "Residual Analysis in the Grouped Data and Censored Normal Linear Model," *Journal of Econometrics*, 34, 33–62.

Choi, I. (2001). "Unit Root Tests for Panel Data," *Journal of International Money and Finance*, 20: 249–272.

Christiano, L. J., M. Eichenbaum, C. L. Evans (1999). "Monetary Policy Shocks: What Have We Learned and to What End?" Chapter 2 in J. B. Taylor and M. Woodford, (eds.), *Handbook of Macroeconomics, Volume 1A*, Amsterdam: Elsevier Science Publishers B.V.

Christiano, Lawrence J. and Terry J. Fitzgerald (2003). "The Band Pass Filter," *International Economic Review*, 44(2), 435-465.

Cleveland, William S. (1993). *Visualizing Data*, Summit, NJ: Hobart Press.

Cleveland, William S. (1994). *The Elements of Graphing Data*, Summit, NJ: Hobart Press.

Conover, W. J. (1980). *Practical Nonparametric Statistics*, 2nd edition, New York: John Wiley & Sons.

Conover, W. J., M. E. Johnson and M. M. Johnson (1981). "A Comparative Study of Tests for Homogeneity of Variance with Applications to the Outer Continental Shelf Bidding Data," *Technometrics*, 23, 351–361.

Csörgö, Sandor and Julian Faraway (1996). "The Exact and Asymptotic Distributions of Cramer-von Mises Statistics," *Journal of the Royal Statistical Society, Series B*, 58, 221-234.

D'Agostino and Michael A. Stephens, (eds.) (1986). *Goodness-of-Fit Techniques*. New York: Marcel A. Deckker.

Dallal, Gerard E. and Leland Wilkinson (1986). "An Analytic Approximation to the Distribution of Lilliefor's Test Statistic For Normality," *The American Statistician*, 40(4), 294-296.

Davidson, Russell and James G. MacKinnon (1989). "Testing for Consistency using Artificial Regressions," *Econometric Theory*, 5, 363–384.

Davidson, Russell and James G. MacKinnon (1993). *Estimation and Inference in Econometrics*, Oxford: Oxford University Press.

Davis, Charles S., and Michael A. Stephens (1989). "Empirical Distribution Function Goodness-of-Fit Tests," *Applied Statistics*, 38(3), 535-582.

Davis, Peter (2002). "Estimating Multi-way Error Components Models with Unbalanced Data Structures," *Journal of Econometrics*, 106, 67-95.

Dezhbaksh, Hashem (1990). "The Inappropriate Use of Serial Correlation Tests in Dynamic Linear Models," *Review of Economics and Statistics*, 72, 126–132.

Dickey, D.A. and W.A. Fuller (1979). "Distribution of the Estimators for Autoregressive Time Series with a Unit Root," *Journal of the American Statistical Association*, 74, 427–431.

Ding, Zhuanxin, C. W. J. Granger, and R. F. Engle (1993). "A Long Memory Property of Stock Market Returns and a New Model," *Journal of Empirical Finance*, 1, 83–106.

Doan, Thomas, Robert B. Litterman, and Christopher A. Sims (1984). "Forecasting and Conditional Projection using Realistic Prior Distributions," *Econometric Reviews*, 3, 1–100.

Doornik, Jurgen A. and Henrik Hansen (1994). "An Omnibus Test for Univariate and Multivariate Normality," manuscript.

Doornik, Jurgen A. (1995). "Testing General Restrictions on the Cointegrating Space," manuscript.

Durbin, J. (1970). *Distribution Theory for Tests Based on the Sample Distribution Function*. SIAM: Philadelphia.

Dyer, D. D. and J. P. Keating (1980). "On the Determination of Critical Values for Bartlett's Test," *Journal of the American Statistical Association*, 75, 313–319.

Elliott, Graham, Thomas J. Rothenberg and James H. Stock (1996). "Efficient Tests for an Autoregressive Unit Root," *Econometrica* 64, 813-836.

Engle, Robert F. (1982). "Autoregressive Conditional Heteroskedasticity with Estimates of the Variance of U.K. Inflation," *Econometrica,* 50, 987–1008.

Engle, Robert F. (1984). "Wald, Likelihood Ratio, and Lagrange Multiplier Tests in Econometrics," Chapter 13 in Z. Griliches and M. D. Intriligator (eds.), *Handbook of Econometrics, Volume 2*, Amsterdam: Elsevier Science Publishers B.V.

Engle, Robert F. and C. W. J. Granger (1987). "Co-integration and Error Correction: Representation, Estimation, and Testing," *Econometrica,* 55, 251–276.

Engle, Robert F. and K. F. Kroner (1995). "Multivariate Simultaneous Generalized ARCH," *Econometric Theory,* 11, 122-150.

Engle, R. and G. J. Lee (1999). "A Permanent and Transitory Component Model of Stock Return Volatility," in R. Engle and H. White, (eds.), *Cointegration, Causality, and Forecasting: A Festschrift in Honor of Clive W. J. Granger*, 475-497, Oxford: Oxford University Press

Engle, Robert F., David M. Lilien, and Russell P. Robins (1987). "Estimating Time Varying Risk Premia in the Term Structure: The ARCH-M Model," *Econometrica,* 55, 391–407.

Engle, Robert F. and Victor K. Ng (1993). "Measuring and Testing the Impact of News on Volatility," *Journal of Finance*, 48, 1022–1082.

Engle, Robert F. and Mark W. Watson (1987). "The Kalman filter: Applications to Forecasting and Rational-Expectations Models," Chapter 7 in Truman F. Bewley (ed.), *Advances in Econometrics—Fifth World Congress*, Volume 1, Cambridge: Cambridge University Press.

Evans, M., N. Hastings, and B. Peacock (1993). *Statistical Distributions*, 2nd edition, New York: John Wiley & Sons.

Fahrmeir, Ludwig, and Gerhard Tutz (1994). *Multivariate Statistical Modelling Based on Generalized Linear Models*, New York: Springer Verlag.

Fair, Ray C. (1970). "The Estimation of Simultaneous Equation Models With Lagged Endogenous Variables and First Order Serially Correlated Errors," *Econometrica*, 38, 507–516.

Fair, Ray C. (1978). "A Theory of Extramarital Affairs," *Journal of Political Economy*, 86, 45–61.

Fair, Ray C. (1984). *Specification, Estimation, and Analysis of Macroeconometric Models*, Cambridge, MA: Harvard University Press.

Fama, Eugene F. and Michael R. Gibbons (1982). "Inflation, Real Returns, and Capital Investment," *Journal of Monetary Economics*, 9, 297–323.

Fan, J. and Gijbels, I. (1996). *Local Polynomial Modelling and its Applications*, London: Chapman & Hall.

Fan, J. and J. S. Marron (1994). "Fast Implementations of Nonparametric Curve Estimators," *Journal of Computational and Graphical Statistics*, 3, 35–56.

Fisher, R. A. (1932). *Statistical Methods for Research Workers, 4th Edition*, Edinburgh: Oliver & Boyd.

Glosten, L. R., R. Jaganathan, and D. Runkle (1993). "On the Relation between the Expected Value and the Volatility of the Normal Excess Return on Stocks," *Journal of Finance*, 48, 1779–1801.

Godfrey, L. G. (1988). *Specification Tests in Econometrics*, Cambridge: Cambridge University Press.

Goldberger, Arthur S. (1991). *A Course in Econometrics*, Cambridge, MA: Harvard University Press.

Gourieroux, C., A. Monfort, and C. Trognon (1984a). "Pseudo-Maximum Likelihood Methods: Theory," *Econometrica*, 52, 681–700.

Gourieroux, C., A. Monfort, and C. Trognon (1984b). "Pseudo-Maximum Likelihood Methods: Applications to Poisson Models," *Econometrica*, 52, 701–720.

Gourieroux, C., A. Monfort, E. Renault, and A. Trognon (1987). "Generalized Residuals," *Journal of Econometrics*, 34, 5–32.

Granger, C. W. J. (1969). "Investigating Causal Relations by Econometric Models and Cross-Spectral Methods," *Econometrica*, 37, 424–438.

Grasa, Antonio Aznar (1989). *Econometric Model Selection: A New Approach*, Dordrecht: Kluwer Academic Publishers.

Greene, William H. (1997). *Econometric Analysis*, 3rd Edition, Upper Saddle River, NJ: Prentice Hall.

Gujarati, Damodar N. (1995). *Basic Econometrics*, 3rd Edition, New York: McGraw-Hill.

Hamilton, James D. (1994a). *Time Series Analysis*, Princeton University Press.

Hamilton, James D. (1994b). "State Space Models," Chapter 50 in Robert F. Engle and Daniel L. McFadden (eds.), *Handbook of Econometrics, Volume 4*, Amsterdam: Elsevier Science B.V.

Hardi, Kaddour (2000). "Testing for Stationarity in Heterogeneous Panel Data," *Econometric Journal*, 3, 148–161.

Härdle, Wolfgang (1991). *Smoothing Techniques with Implementation in S*, New York: Springer Verlag.

Harrison, D. and D. L. Rubinfeld (1978). "Hedonic Housing Prices and the Demand for Clean Air," *Journal of Environmental Economics and Management*, 5, 81-102.

Harvey, Andrew C. (1987). "Applications of the Kalman Filter in Econometrics," Chapter 8 in Truman F. Bewley (ed.), *Advances in Econometrics—Fifth World Congress*, Volume 1, Cambridge: Cambridge University Press.

Harvey, Andrew C. (1989). *Forecasting, Structural Time Series Models and the Kalman Filter,* Cambridge: Cambridge University Press.

Harvey, Andrew C. (1990). *The Econometric Analysis of Time Series*, 2nd edition, Cambridge, MA: MIT Press.

Harvey, Andrew C. (1993). *Time Series Models*, 2nd edition, Cambridge, MA: MIT Press.

Hayashi, Fumio. (2000). *Econometrics*, Princeton, NJ: Princeton University Press.

Hausman, Jerry A. (1978). "Specification Tests in Econometrics," *Econometrica*, 46, 1251–1272.

Hodrick, R. J. and E. C. Prescott (1997). "Postwar U.S. Business Cycles: An Empirical Investigation," *Journal of Money, Credit, and Banking*, 29, 1–16.

Holzer, H., R. Block, M. Cheatham, and J. Knott (1993), "Are Training Subsidies Effective? The Michigan Experience," *Industrial and Labor Relations Review,* 46, 625-636.

Hosmer, David W. Jr. and Stanley Lemeshow (1989). *Applied Logistic Regression*, New York: John Wiley & Sons.

Hyndman, Rob J. and Yanan Fan (1996). "Sample Quantiles in Statistical Packages," *The American Statistician*, 50(4), 361-365.

Im, K. S., Pesaran, M. H., and Y. Shin (2003). "Testing for Unit Roots in Heterogeneous Panels," *Journal of Econometrics*, 115, 53–74.

Jain, Raj and Imrich Chlamtac (1985). "The P2 Algorithm for Dynamic Calculation of Quantiles and Histograms Without Storing Observations," *Communications of the ACM*, 28(10), 1076–1085.

Johansen, Søren and Katarina Juselius (1990). "Maximum Likelihood Estimation and Inferences on Cointegration—with applications to the demand for money," *Oxford Bulletin of Economics and Statistics*, 52, 169–210.

Johansen, Søren (1991). "Estimation and Hypothesis Testing of Cointegration Vectors in Gaussian Vector Autoregressive Models," *Econometrica*, 59, 1551–1580.

Johansen, Søren (1995). *Likelihood-based Inference in Cointegrated Vector Autoregressive Models*, Oxford: Oxford University Press.

Johnson, Norman L. and Samuel Kotz (1969). *Discrete Distributions*, Boston: Houghton Mifflin.

Johnson, Norman L. and Samuel Kotz (1970). *Continuous Univariate Distributions 1 & 2*, Boston: Houghton Mifflin.

Johnston, Jack and John Enrico DiNardo (1997). *Econometric Methods*, 4th Edition, New York: McGraw-Hill.

Judge, George G., W. E. Griffiths, R. Carter Hill, Helmut Lütkepohl, and Tsoung-Chao Lee (1985). *The Theory and Practice of Econometrics*, *2nd edition*, New York: John Wiley & Sons.

Kelejian, H. H. (1982). "An Extension of a Standard Test for Heteroskedasticity to a Systems Framework," *Journal of Econometrics*, 20, 325-333.

Kennan, John (1985). "The Duration of Contract Strikes in U.S. Manufacturing," *Journal of Econometrics*, 28, 5–28.

Kincaid, David, and Ward Cheney (1996). *Numerical Analysis, 2nd edition*, Pacific Grove, CA: Brooks/Cole Publishing Company.

Knuth, D. E. (1997). *The Art of Computer Programming, Volume 2, Semi-numerical Algorithms, 3rd edition*, Reading, MA: Addison-Wesley Publishing Company. *Note: the C implementation of the lagged Fibonacci generator is described in the errata to the 2nd edition, downloadable from Knuth's web site.*

Koopman, Siem Jan, Neil Shephard, and Jurgen A. Doornik (1999). "Statistical Algorithms for Models in State Space using SsfPack 2.2," *Econometrics Journal*, 2(1), 107-160.

Kwiatkowski, Denis, Peter C. B. Phillips, Peter Schmidt & Yongcheol Shin (1992). "Testing the Null Hypothesis of Stationary against the Alternative of a Unit Root," *Journal of Econometrics,* 54, 159-178.

LeBaron, Blake (1997). "A Fast Algorithm for the BDS statistic," *Studies in Nonlinear Dynamics and Econometrics*, 2(2), 53–59.

L'Ecuyer, P. (1999). "Good Parameters and Implementations for Combined Multiple Recursive Random Number Generators," *Operations Research*, 47(1), 159-164

Levene, H. (1960). "Robust Tests for the Equality of Variances," in I. Olkin, S. G. Ghurye, W. Hoeffding, W. G. Madow, and H. B. Mann (eds.), *Contribution to Probability and Statistics*, Palo Alto, CA: Stanford University Press.

Levin, A., Lin, C. F., and C. Chu (2002). "Unit Root Tests in Panel Data: Asymptotic and Finite-Sample Properties," *Journal of Econometrics*, 108, 1–24.

Lewis, Peter A. W. (1961). "Distribution of the Anderson-Darling Statistic," *Annals of Mathematical Statistics*, 32, 1118-1124.

Ljung, G. and G. Box (1979). "On a Measure of Lack of Fit in Time Series Models," *Biometrika*, 66, 265–270.

Lütkepohl, Helmut (1991). *Introduction to Multiple Time Series Analysis*, New York: Springer-Verlag.

MacKinnon, James G. (1991). "Critical Values for Cointegration Tests," Chapter 13 in R. F. Engle and C. W. J. Granger (eds.), *Long-run Economic Relationships: Readings in Cointegration*, Oxford: Oxford University Press.

MacKinnon, James G. (1996). "Numerical Distribution Functions for Unit Root and Cointegration Tests," *Journal of Applied Econometrics*, 11, 601-618.

MacKinnon, James G., Alfred A. Haug, and Leo Michelis (1999), "Numerical Distribution Functions of Likelihood Ratio Tests For Cointegration," *Journal of Applied Econometrics*, 14, 563-577.

Maddala, G. S. and S. Wu (1999). "A Comparative Study of Unit Root Tests with Panel Data and A New Simple Test," *Oxford Bulletin of Economics and Statistics*, 61, 631–52.

Marron, J. S. and D. Nolan (1989). "Canonical Kernels for Density Estimation," *Statistics and Probability Letters*, 7, 191–195.

Marsaglia, G. (1993). "Monkey Tests for Random Number Generators," *Computers and Mathematics with Applications*, 9, 1-10.

Matsumoto, M. and T. Nishimura (1998). "Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator," *ACM Transactions on Modeling and Computer Simulation*, 8(1), 3-30.

McCallum, Bennett T. (1989). "Real Business Cycle Models," Chapter 1 in Robert J. Barro (ed.), *Modern Business Cycle Theory*, Cambridge, MA: Harvard University Press.

McCullagh, P. and J. A. Nelder (1989). *Generalized Linear Models, 2nd Edition*, London: Chapman & Hall.

McDonald, J. and R. Moffitt (1980). "The Uses of Tobit Analysis," *Review of Economic and Statistics*, 62, 318–321.

Nelson, Daniel B. (1991). "Conditional Heteroskedasticity in Asset Returns: A New Approach," *Econometrica*, 59, 347–370.

Neter, John, Michael H. Kutner, Christopher J. Nachtsheim, and Willian Wasserman (1996). *Applied Linear Statistical Models*, 4th Edition. Chicago: Times Mirror Higher Education Group, Inc. and Richard D. Irwin, Inc.

Newey, Whitney and Kenneth West (1987a). "Hypothesis Testing with Efficient Method of Moments Estimation," *International Economic Review*, 28, 777–787.

Newey, Whitney and Kenneth West (1987b). "A Simple Positive Semi-Definite, Heteroskedasticity and Autocorrelation Consistent Covariance Matrix," *Econometrica*, 55, 703–708.

Newey, Whitney and Kenneth West (1994). "Automatic Lag Selection in Covariance Matrix Estimation," *Review of Economic Studies*, 61, 631-653.

Ng, Serena and Pierre Perron (2001). "Lag Length Selection and the Construction of Unit Root Tests with Good Size and Power," *Econometrica*, 69(6), 1519-1554.

Osterwald-Lenum, Michael (1992). "A Note with Quantiles of the Asymptotic Distribution of the Maximum Likelihood Cointegration Rank Test Statistics," *Oxford Bulletin of Economics and Statistics*, 54, 461–472.

Pagan, A. and F. Vella (1989). "Diagnostic Tests for Models Based on Individual Data: A Survey," *Journal of Applied Econometrics*, 4, S29–S59.

Pesaran, M. Hashem and Yongcheol Shin (1998). "Impulse Response Analysis in Linear Multivariate Models," *Economics Letters*, 58, 17-29.

Phillips, Peter C. B. and S. Ouliaris (1990). "Asymptotic Properties of Residual Based Tests for Cointegration," *Econometrica*, 58, 165–193.

Phillips, P.C.B. and P. Perron (1988). "Testing for a Unit Root in Time Series Regression," *Biometrika*, 75, 335–346.

Pindyck, Robert S. and Daniel L. Rubinfeld (1991). *Econometric Models and Economic Forecasts*, 3rd edition, New York: McGraw-Hill.

Papke, L. E. (1994). "Tax Policy and Urban Development: Evidence From the Indiana Enterprise Zone Program," *Journal of Public Economics*, 54, 37-49.

Powell, J. L. (1986). "Symmetrically Trimmed Least Squares Estimation for Tobit Models," *Econometrica*, 54, 1435–1460.

Prescott, Edward C. (1986). "Theory Ahead of Business-Cycle Measurement," *Carnegie-Rochester Conference Series on Public Policy*, 25, 11–44.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1992). *Numerical Recipes in C, 2nd edition*, Cambridge University Press.

Quandt, Richard E. (1983). "Computational Problems and Methods," Chapter 12 in Z. Griliches and M. D. Intriligator (eds.), *Handbook of Econometrics, Volume 1*, Amsterdam: Elsevier Science Publishers B.V..

Ramsey, J. B. (1969). "Tests for Specification Errors in Classical Linear Least Squares Regression Analysis," *Journal of the Royal Statistical Society, Series B*, 31, 350–371.

Ramsey, J. B. and A. Alexander (1984). "The Econometric Approach to Business-Cycle Analysis Reconsidered," *Journal of Macroeconomics*, 6, 347–356.

Rao, P. and Z. Griliches (1969). "Small Sample Properties of Several Two-Stage Regression Methods in the Context of Auto-Correlated Errors," *Journal of the American Statistical Association*, 64, 253–272.

Ravn, Morten O. and Harald Uhlig (2002). "On Adjusting the Hodrick-Prescott Filter for the Frequency of Observations," *Review of Economics and Statistics*, 84, 371-375

Said, Said E. and David A. Dickey (1984). "Testing for Unit Roots in Autoregressive Moving Average Models of Unknown Order," *Biometrika*, 71, 599–607.

Schwert, W. (1989). "Stock Volatility and Crash of '87," *Review of Financial Studies,* 3, 77–102.

Sheskin, David J. (1997). *Parametric and Nonparametric Statistical Procedures,* Boca Raton: CRC Press.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis*, London: Chapman & Hall.

Simonoff, Jeffrey S. (1996). *Smoothing Methods in Statistics*, New York: Springer-Verlag.

Sims, Chris (1980). "Macroeconomics and Reality," *Econometrica*, 48, 1-48.

Sims, Chris (1986). "Are Forecasting Models Usable for Policy Analysis?" *Quarterly Review of the Federal Reserve Bank of Minneapolis*, 2-16.

Sokal, Robert R. and F. James Rohlf (1995). *Biometry.* New York: W. H. Freeman and Company.

Stephens, Michael A. (1986). "Tests Based on EDF Statistics," in *Goodness-of-Fit Techniques*, Ralph B. D'Agostino and Michael A. Stephens, (eds.). New York: Marcel A. Deckker, 97-193.

Tauchen, George (1986). "Statistical Properties of Generalized Method-of-Moments Estimators of Structural Parameters Obtained From Financial Market Data," *Journal of Business & Economic Statistics*, 4, 397–416.

Taylor, S. (1986). *Modeling Financial Time Series*, New York: John Wiley & Sons.

Temme, Nico M. (1996). *Special Functions: An Introduction to the Classical Functions of Mathematical Physics*, New York: John Wiley & Sons.

Thisted, Ronald A. (1988). *Elements of Statistical Computing*, New York: Chapman and Hall.

Urzua, Carlos M. (1997). "Omnibus Tests for Multivariate Normality Based on a Class of Maximum Entropy Distributions," in *Advances in Econometrics*, Volume 12, Greenwich, Conn.: JAI Press, 341-358.

Wansbeek, Tom, and Arie Kapteyn (1989). "Estimation of the Error Components Model with Incomplete Panels," *Journal of Econometrics,* 41, 341-361.

Watson, Mark W. and Robert F. Engle (1983). "Alternative Algorithms for the Estimation of Dynamic Factor, MIMIC and Varying Coefficient Regression Models," *Journal of Econometrics*, 23, 385–400.

White, Halbert (1980)."A Heteroskedasticity-Consistent Covariance Matrix and a Direct Test for Heteroskedasticity," *Econometrica*, 48, 817–838.

White, Halbert (1982). "Maximum Likelihood Estimation of Misspecified Models," *Econometrica*, 50, 1–26.

Wooldridge, Jeffrey M. (1990). "Quasi-Likelihood Methods for Count Data," Chapter 8 in M. Hashem Pesaran and P. Schmidt (eds.) *Handbook of Applied Econometrics, Volume 2*, Malden, MA: Blackwell, 352–406.

Wooldridge, Jeffrey M. (1997). "A Note on the Lagrange Multiplier and F-statistics for Two Stage Least Squares Regressions," *Economics Letters*, 34, 151-155.

Wooldridge, Jeffrey M. (2000). *Introductory Econometrics: A Modern Approach*, South-Western College Publishing.

Wooldridge, Jeffrey M. (2002). *Econometric Analysis of Cross Section and Panel Data*, Cambridge, MA: The MIT Press.

Zakoïan, J. M. (1994). "Threshold Heteroskedastic Models," *Journal of Economic Dynamics and Control*, 18, 931-944.

Index

Symbols

.DB? files *266*
.EDB file *262*
.RTF file *437*
.WF1 file *62*
?
  pool cross section identifier *830*
  wildcard versus pool identifier *948*
@all *97*
@cellid *879*
@crossid *878*
@expand *466*
@first *97*
@firstmax *884*
@firstmin *884*
@ingrp *830*
@last *97*
  with smpl *97*
@lastmax *884*
@lastmin *884*
@map *173*
@obsid *880*
@obsnum
  Panel
  observation numbering *880*
@unmap *174*
@unmaptxt *174*
~, in backup file name *62*, *939*

Numerics

2sls (two-stage least squares) *473*
  with pooled data *867*
3sls (three-stage least squares) *697*, *716*

A

Abort key *21*
Active window *18*, *79*
Add factor *777*
Add text to graph *421*
Adjusted R-squared
  for regression *451*
Advanced database query *280*
AIC *971*
Akaike criterion *453*, *971*
  for equation *453*
Alias *779*
  database *276*
  OBALIAS.INI file *287*
  object *286*
Almon lag *462*
Alpha series *153*
  declaring and creating *154*
  maximum length *155*, *942*
  truncation *155*, *942*
Analysis of variance *318*
  by ranks *321*
Analytic derivatives *966*
And operator *98*, *133*
Anderson-Darling test *323*
Andrews test *631*, *668*
ANOVA *318*
  by ranks *321*
AR specification
  estimation *497*
  forecast *558*
  in 2SLS *477*
  in ARIMA models *501*
  in nonlinear 2SLS *486*
  in nonlinear least squares *485*
  in pool *847*
  in systems *699*
AR(1)
  coefficient *494*
  Durbin-Watson statistic *494*
  estimation *498*
AR(p) *494*
  estimation *498*
ARCH *601*
  correlogram test *580*
  LM test *582*
ARCH-M *603*
Area graph *310*
AREMOS data *293*
ARIMA models *501*

Box-Jenkins approach *502*
diagnostic checking *512*
difference operator *504*
identification *502*
specification *503*
starting values *509*
ARMA terms *505*
in models *809*
seasonal *506*
testing *581*
using state spaces models for *762*
Artificial regression *583*, *594*
ASCII
file export *115*
file import *112*, *128*
ASCII file
open as workfile *54*
Asymptotic test *569*
Augmented Dickey-Fuller test *522*
See also Unit root tests.
Augmented regression *587*
Autocorrelation *452*
Autocorrelation function *326*
Auto-search
database *277*
Auto-series
generate new series *141*, *274*
in estimation *145*
in groups *144*
in regression *274*
with database *274*
Autospec *758*
Auto-updating series
and databases *152*
converting to ordinary series *152*
Auxiliary regression *581*, *582*
Average log likelihood *626*

B

Backcast
in GARCH models *606*
MA terms *510*
Backup file option *62*
Backup files *939*
Balanced data *835*
Balanced sample *846*
Bandwidth *397*
Andrews *719*
bracket option *398*, *404*
in kernel fit *406*
in nearest neighbor fit *403*
Newey-West (fixed) *718*
Newey-West (variable) *719*
selection in GMM *704*, *718*
Bar graph *310*
Bartlett kernel *718*
Bartlett test *322*
BDS test *329*
Berndt-Hall-Hall-Hausman (BHHH). See Optimization algorithms.
Bias proportion *554*
Binary dependent variable *621*
error messages *627*
fitted index *634*
interpretation of coefficient *625*
log likelihood *622*
Binary Estimation
perfect predictor *628*
Binary file *54*
Binning option
classifications *313*, *382*
in density estimation *398*, *407*
Binomial sign test *317*
Box-Cox transformation *401*
Box-Jenkins *502*
Boxplots *409*
for series by classification *410*
for series in group *412*
Bracket bandwidth option
in kernel density *398*
in kernel fit *407*
in nearest neighbor fit *404*
Break key *21*
Breusch-Godfrey test *495*, *581*
Brown-Forsythe test *322*
By-group statistics *312*, *886*, *893*

C

C
coef vector *444*
constant in regression *444*
Cache *301*

Cancel
keystroke *21*
Categorical regressor stats *629*, *649*
Causality
Granger's test *389*
Censored dependent variable *644*
fitted index *650*
goodness-of-fit tests *652*
interpretation of coefficient *649*
log likelihood *645*
scale factor *649*
specifying the censoring point *646*
Census X11
historical *344*
limitations *345*
using X12 *338*
Census X12 *337*
seasonal adjustment options *338*
Change default directory *71*
Chi-square
independence test in tabulation *383*
statistic for Wald test *574*
test for independence in n-way table *384*
test for the median *321*
Cholesky factor
in VAR impulse responses *730*
in VAR normality test *727*
Chow test
breakpoint *584*
forecast *585*
n-step forecast *591*
one-step forecast *591*
Classification table
binary models *629*
ordered models *642*
sensitivity *630*
specificity *630*
Cleveland subsampling *403*
Click(ing) *18*
Clipboard *428*
Close
EViews *25*
object *938*
window *19*
Cochrane-Orcutt *478*, *500*
Coef (coefficient vector)
default *444*
update from equation *457*
Coefficient
common (pool) *847*
covariance matrix *455*
covariance matrix of estimates *456*
cross-section specific (pool) *847*
estimated from regression *449*
maximum number in default *642*
recursive estimates *592*
restrictions *447*
setting initial values *483*, *951*
standard error *450*
Coefficient restriction test *572*
Coefficient uncertainty *796*, *805*, *812*
Cointegration test *739*
Collinearity *461*
Column width *433*
Command window *23*
Comments *82*
Common sample *135*
Comparison operators
with missing values *135*
Compatibility *4*
Component GARCH *615*
Conditional independence *384*
Conditional standard deviation
graph *610*
Conditional variance *599*, *601*, *602*
forecast *611*
in the mean equation *603*
make series *611*
Confidence ellipses *570*
Confidence interval
ellipses *570*
for forecast *552*
for stochastic model solution *811*
Constant
in equation *444*, *450*
in ordered models *640*
Contingency coefficient *384*
Convergence criterion *953*
in nonlinear least squares *483*, *487*
in pool estimation *850*
Copy
and paste *83*
data *114*
data cut-and-paste *107*

database *287*
graph *428*
object *82*
table to clipboard *437*
to and from database *270*
Copy and paste *428*, *437*
Correlation *387*
Correlogram *329*, *387*, *495*
autocorrelation function *326*
cross *387*
partial autocorrelation function *327*
Q-statistic *328*
squared residuals *580*, *610*
Count models *658*
negative binomial (ML) *659*
Poisson *659*
quasi-maximum likelihood *660*
Covariance *387*
Covariance matrix
HAC (Newey-West) *473*
heteroskedasticity consistent of estimated coefficients *472*
of estimated coefficients *455*
Covariance proportion *554*
Cramer's V *384*
Cramer's V *384*
Cramer-von Mises test *323*
Create
database *263*
group *93*, *145*
object *77*
series *88*, *137*
text object *438*
Cross correlation *387*
Cross correlogram *387*
Cross-equation
coefficient restriction *696*, *699*
correlation *697*, *698*
weighting *696*
Cross-section
identifiers in pool *828*
Cross-section specific series *829*
CSV *438*
Cumulative distribution *391*
Blom option *393*
Ordinary option *393*
Rankit option *393*
Tukey option *393*
Van der Waerden option *393*
CUSUM
sum of recursive residuals test *589*
sum of recursive squared residuals test *590*

D

Data
cut and paste *109*, *114*
export *114*
import *109*
keyboard entry *107*
pool *832*
Database
alias *276*
auto-search *277*
auto-series *275*
cache *301*
copy *287*
copy objects *270*
create *263*
data storage precision *940*
default *266*
default in search order *277*
delete *287*
delete objects *272*
display all objects *264*
export *269*
fetch objects *268*
field *280*
frequency in query *282*
group storage options *940*
link *300*
link options *303*
match operator in query *282*
open *263*
packing *288*
queries *277*
rebuild *289*
registry *940*
rename *287*
rename object *271*
repair *289*
sharing violation *265*
statistics *288*
store objects *267*
test integrity *289*
using auto-updating series with *152*

window *263*
Database registry *275*
Date series *160*
Dated data table *365*
formatting options *371*
frequency conversion *369*
row options *371*
table options *366*
transformation methods *368*
Dates
default display format *940*
global options *939*
match merging using *183*
Default
database *24*, *266*
database in search order *277*
directory *24*, *938*
set directory *71*
setting global options *937*
update directory *71*
Delete *84*
database *287*
objects from database *272*
observation in series *93*
series using pool *844*
Demonstration
estimation *36*
forecasting *42*
getting data into EViews *27*
specification test *38*
Dependent variable
no variance in binary models *628*
Derivatives *954*, *966*
description *966*
saving in series *969*
Description
field in database query *284*
Descriptive statistics
balanced sample (pool) *840*
by classification *312*
by group *312*
common sample (group) *379*
common sample (pool) *840*
cross-section specific *840*
for a series *314*
group *379*
individual samples (group) *379*
individual samples (pool) *840*
stacked data *840*
time period specific *840*
Deselect all *77*
Dialog box *21*
Dickey-Fuller test *522*
See also Unit root tests.
Difference from moving-average *350*
Difference operator *134*, *504*
seasonal *134*, *504*
Display filter *58*, *207*
Display format
for groups *94*
Display name
field in database query *284*
Distribution
empirical distribution function tests *323*
tests of *323*
Distribution graphs *322*, *391*
cumulative distribution *391*
kernel density *323*, *396*
QQ-plot *323*, *393*
quantile *392*
survivor *392*
Doornik and Hansen factorization matrix *727*
Drag(ging) *18*
text in graph *423*
DRI
frequency *303*
queries *304*
DRI database
DRIpro *300*
illegal names *303*
object alias *286*
shadowing of object names *286*
troubleshooting *305*
DRIBase database *291*
DRIPro link *291*
Dummy variables
as binary dependent variable *621*
as censoring point in estimation *647*
automatic creation in estimation *466*
generating pool series dummies *838*
using @GROUP to create pool dummies *838*
Durbin-Watson statistic *494*
for regression *452*
lagged dependent variable *495*

Dynamic forecasting *556*

E

Easy query *278*
EcoWin database *295*
Edit
  group *95*
  series *91*, *364*
  table *432*
Elliot, Rothenberg, and Stock point optimal test *525*
  See also Unit root tests.
Empirical distribution function tests *323*
End field *283*
Endogeneity *594*
Endogenous variable *473*
Enterprise Edition *292*, *296*
Epanechnikov kernel *397*, *406*
Equality tests *318*
  mean *318*
  median *320*
  variance *321*
Equation
  automatic dummy variables in *466*
  coefficient covariance matrix *455*
  coefficient covariance scalar *454*
  coefficient standard error vector *454*, *455*
  coefficient t-statistic scalar *454*
  coefficient t-statistic vector *455*
  coefficient vector *455*
  create *443*
  regression coefficients *449*
  regression summary statistics *451*
  results *449*
  retrieve previously estimated *458*
  saved results *454*, *457*
  scalar results *454*
  specification *444*
  specify by formula *446*
  specify by list *444*
  specify with non-default coefs *447*
  specify with restrictions *447*
  specify without dependent variable *446*
  specifying a constant *444*
  store *458*
  text representation *455*
  t-statistic *450*
  vector and matrix results *455*
Error bar graph *375*, *418*
Estimation
  AR specification *497*
  auto-series *145*
  binary dependent variable *623*
  censored models *645*
  collinearity *460*
  convergence problems *954*
  count models *658*
  derivative computation options *954*
  failure to improve *954*
  for pool *845*
  missing data *448*
  near singular matrix problems *953*
  nonlinear least squares *480*
  options *951*
  ordered models *639*
  output *449*
  problems in convergence *952*
  residuals from equation *457*
  sample *448*
  sample adjustment *448*
  single equation methods *447*
  state space *767*
  systems *703*
  truncated models *655*
  two-stage least squares *473*
  weighted least squares *471*
Evaluate
  precedence *130*
EViews
  Enterprise Edition *293*
EViews Databases *261*
EViews Enterprise Edition *292*, *296*
Excel file
  opening as workfile *54*
  opening as workfile demo *27*
Excel files
  importing into workfile *109*
Exogenous variable *473*
Exogenous variable uncertainty *805*, *812*
Expectation-prediction table
  binary models *629*
  ordered models *642*
Expectations consistency in models *807*
Expected dependent variable

censored models *651*
truncated models *656*
Expected latent variable
censored models *650*
truncated models *656*
Exponential GARCH (EGARCH) *613*
Exponential smoothing *356*
double *352*
Holt-Winters additive *354*
Holt-Winters multiplicative *353*
Holt-Winters no-seasonal *354*
single *352*
Export data *114*
from pool objects *844*
to ASCII files *115*
to spreadsheet files *115*
Export database *269*
Expression *129*
for database fields *281*
parentheses *130*
Extended search in models *815*
Extreme value
binary model *624*

F

Fair-Taylor model solution *806*
FAME database *292*
Fetch
from database *268*
from pool *844*
Fields in database *280*
description *284*
display_name *284*
end *283*
expressions *281*
freq *282*
history *284*
last_update *283*
last_write *283*
name *281*
remarks *284*
source *284*
start *283*
type *282*
units *284*
File
save table to *438*
Files
default locations *938*
First derivative methods *957*
Fitted index
binary models *634*
censored models *650*
truncated models *656*
Fitted probability
binary models *633*
Fitted values
of equation *455*
Fixed effects *863*
Fixed effects test *926*
Fixed variance parameter
negative binomial QML *662*
normal QML *661*
Font options
text in graph *421*, *422*
Fonts
defaults *937*
Forecast
AR specification *558*
binary models *633*
by exponential smoothing *356*
censored models *650*
Chow test *585*
conditional variance *611*
count models *662*
dynamic *556*, *757*
equations with formula *561*
error *551*
evaluation *553*
expressions and auto-updating series *561*
fitted values *550*
from estimated equation *543*
interval *552*
MA specification *558*
n-step ahead *756*
n-step test *591*
one-step test *591*
ordered models *643*
out-of-sample *549*
smoothed *757*
standard error *552*, *564*
static *557*
structural *557*
truncated models *656*
variance *551*

with AR errors 558
Foreign data
open as workfile 27, 53
Formula
forecast 561
implicit assignment 139
normalize 140
specify equation by 446
Forward solution for models 805
Freeze 84
create graph from view 415
Freq
field in database query 282
Frequency conversion 83, 115, 939
DRI database 303
in dated data table 369
propagate NAs 117
undated series 119
using links 187
F-statistic 574, 577
for regression 453
F-test
for variance equality 321
Full information maximum likelihood 698

G

GARCH 601
ARCH-M model 603
asymmetric component model 616
backcasting 606
component models 615
exponential GARCH (EGARCH) 613
GARCH(1,1) model 601
GARCH(p,q) model 603
initialization 606
mean equation 605
multivariate 692
robust standard errors 607
threshold (TARCH) 613
Gauss file 54
Gauss-Newton 958
Gauss-Seidel algorithm 814, 959
Generalized error distribution 613
Generalized least squares 848
Generalized linear models 667
quasi-likelihood ratio test 662
robust standard errors 667
variance factor 668
Generalized method of moments. See GMM.
Generalized residual
binary models 634
censored models 650
count models 663
ordered models 644
score vector 635
truncated models 656
Generate series 137
by command 140
dynamic assignment 139
for pool 838
implicit formula 139
using samples 138
Geometric moving average 144
GiveWin data 293
GLM (generalized linear model) 667
standard errors 667
Global optimum 953
GLS detrending 524
GMM
bandwidth selection 704
for systems 698
HAC weighting matrix 718
J-statistic 491
kernel options 704
orthogonality condition 488
overidentifying restrictions 491
prewhitening option 705, 720
single equation 488
system 716
White weighting matrix 717
Goldfeld-Quandt 957
Gompit models 624
Goodness-of-fit
adjusted R-squared 451
Andrews test 631, 668
forecast 553
Hosmer-Lemeshow test 631, 668
R-squared 451
Gradients 963
saving in series 965
summary 964
Granger causality 388
test in VAR 724
Graph

aspect ratio *419*
axes control *419*
background color *419*
background printing *419*
border *419*
color settings *419*
combining *416*
coordinates for positioning elements *422*
creating *415*
customizing lines and symbols *420*
drawing lines and shaded areas *423*
error bar *418*
font options *421*
frame fill *419*
grid lines *419*
high-low-open-close *418*
indentation *419*
merging multiple *77*
modifying *416*
multiple graph options *426*
multiple graph positioning *427*
non-consecutive observations *418*
place text in *421*
print in color *428*
printing *427*
remove elements *423*
sample break plotting options *418*
spike *418*
stacked lines and bars *418*
templates *424*
text justification *422*
Grid search *959*
Group *148*
add member *363*
auto-series *144*
create *93*, *145*
creating using wildcards *946*
edit series *364*
editing *95*
element *146*
spreadsheet view *363*
spreadsheet view defaults *942*
Group into bins option *313*, *382*
Groups
display format *94*
Groupwise heteroskedasticity *380*

H

Hannan-Quinn criterion *971*
Hatanaka two step estimator *500*
Hausman test *594*, *928*
Haver Analytics database *293*
Help
help system *25*
World Wide Web *26*
Heteroskedasticity
binary models *637*
groupwise *380*
of known form *469*
of unknown form *471*
White's test *582*
Heteroskedasticity and autocorrelation consistent covariance (HAC) *473*
Heteroskedasticity consistent covariance matrix *472*
High-Low (Open-Close) graphs *375*, *418*
Hildreth-Lu *500*
Histogram *310*, *580*
History
field in database query *284*
Hodrick-Prescott filter *357*
Holt-Winters
additive *354*
multiplicative *353*
no-seasonal *354*
Hosmer-Lemeshow test *631*, *668*
HTML *438*
open page as workfile *54*
save table as web page *438*
Huber/White standard errors *667*
Hypothesis tests
ARCH *582*
Bartlett test *322*
BDS independence *329*
binomial sign test *317*
Brown-Forsythe *322*
chi-square test *321*
Chow breakpoint *584*
coefficient p-value *450*
CUSUM *589*
CUSUM of squares *590*
distribution *323*
F-test *321*

Hausman test *594*
heteroskedasticity *582*
irrelevant or redundant variable *579*
Kruskal-Wallis test *321*
Levene test *322*
multi-sample equality *318*
nonnested *595*
normality *580*
omitted variables *577*
Ramsey RESET *586*
Siegel-Tukey test *322*
single sample *315*
stability test *584*
unit root *329*, *518*
Van der Waerden test *317*, *321*
Wald coefficient restriction test *572*
Wilcoxon signed ranks test *317*

I

Icon *75*
Identification
Box-Jenkins *502*
nonlinear models *487*
Identity
in model *778*
in system *700*
If condition in samples *97*
Import
See also Foreign data *53*
Import data
for pool objects *832*
from ASCII *112*, *120*
from spreadsheet *110*
using a pool object *835*
Impulse response *729*
generalized impulses *731*
See also VAR.
standard errors *730*
structural decomposition *731*
transformation of impulses *730*
user specified impulses *731*
Incorrect functional form *583*, *587*
Independence test *329*
Index
fitted from binary models *634*
fitted from censored models *650*
fitted from truncated models *656*
Individual sample *135*
Information criterion
Akaike *453*, *971*
Hannan-Quinn *971*
Schwarz *453*, *971*
Innovation *493*
Insert
observation *93*
Insertion point *23*
Instrumental variable *473*
for 2SLS with AR specification *478*
for GMM *488*
for nonlinear 2SLS *486*
identification *703*
in systems *700*
order condition *475*
Instrumental variables
with pooled data *867*
Instruments *474*
using PDL specifications *463*
Integer dependent variable *658*
Integrated series *517*
Integrity (database) *289*
Intercept in equation *444*, *450*
Inverted AR roots *500*, *507*
Inverted MA roots *507*
Irrelevant variable test *579*
Iteration
failure to improve message *954*
in models *815*
Iteration option *953*
in nonlinear least squares *483*

J

Jarque-Bera statistic *580*, *611*
in VAR *726*
J-statistic *491*
ordinary equation *491*
panel equation *922*
J-test *596*

K

Kalman filter *755*
Kernel
bivariate fit *405*
choice in HAC weighting *704*, *718*

density estimation *323*, *396*
Kernel function
in density estimation *396*
in kernel regression *406*
Keyboard
data entry *106*
focus option *938*
shortcuts *21*
Keyboard focus *938*
Kolmogorov-Smirnov test *323*
KPSS test *525*
Kruskal-Wallis test *321*
Kullback-Leibler *971*
Kurtosis *312*
Kwiatkowski, Phillips, Schmidt, and Shin test *525*

L

Label *82*
automatic update option *943*
capitalization *82*
pie graph *376*
series *333*
Lag
dynamic assignment *139*
series *133*
Lagged dependent variable
and serial correlation *493*
Durbin-Watson statistic *495*
Lagged series in equation *445*
Lagrange multiplier test *495*
Large sample test *569*
Last_update
field in database query *283*
Last_write
field in database query *283*
Latent variable
binary model *622*
censored models *644*
ordered models *638*
Lead
series *133*
Levene test *322*
Likelihood *452*
Lilliefors test *323*
Limit points *641*
make covariance matrix *643*
make vector *643*
non-ascending *642*
Limited dependent variable *621*
Line drawing *423*
Line graph *310*
Link *177*
create by copy-and-paste *83*
frequency conversion *187*
match merging *178*
Linked equations in models *795*
List
specifying equation by *444*
Ljung-Box Q-statistic *328*
serial correlation test *495*
LM test
ARCH *582*
artificial regression *966*
auxiliary regression *581*
serial correlation *495*, *581*
Load
workfile *63*
Local optimum *953*
Local regression *403*
Local weighting option *404*
LOESS *403*, *405*
Log likelihood
average *626*
censored models *645*
exponential *661*
for binary models *622*
for regression (normal errors) *452*
negative binomial *659*
normal *661*
ordered models *639*
Poisson model *659*
restricted *626*
truncated models *654*
Logical expression *132*
in easy query *279*
Logit models *623*
Long name *81*
for series *333*
LOWESS. See also LOESS
LR statistic *577*, *626*, *652*, *654*, *662*
QLR *666*

M

MA specification
  backcasting *510*
  forecast *558*
  in ARIMA models *501*
  in two stage least squares *479*
Marginal significance level *450*, *569*
Marquardt *958*
Match merge
  using links *178*
Match operator in database query *282*
Maximize window *19*
Maximum likelihood
  full information *698*
  quasi-generalized pseudo-maximum likelihood *665*
  quasi-maximum likelihood *660*
  user specified *671*
McFadden R-squared *626*
Mean *311*
Mean absolute error *553*
Mean absolute percentage error *553*
Measurement equation *754*
Measurement error *473*, *587*
Median *311*
Menu *20*, *80*
  main *20*
  objects *81*
Merge *83*
  graphs *77*
  store option *268*
Metafile
  save graph as Windows metafile. *429*
Micro TSP
  opening workfiles *259*
Minimize window *19*
Missing values *134*
  handling in estimation *448*
  in frequency conversion *117*
  in models *815*
  recoding *137*
  relational comparisons invovlving *135*
  test *136*
Model consistent expectations *807*
Models
  add factors *777*, *802*
  aliasing *779*, *802*
  binding variables *779*
  block structure *798*
  coefficient uncertainty *796*, *805*, *812*
  convergence test *815*
  creating *794*
  definition *695*
  diagnostic messages and iteration history *813*
  dynamic solution *809*
  equation view *796*
  excluding variables *801*
  exogenous variable uncertainty *805*, *812*
  extended search *815*
  Fair-Taylor solution *806*
  fit option for solution *809*
  future values *805*
  Gauss-Seidel solution *959*
  handling of ARMA terms *809*
  initialize excluded variables *814*
  inline equations *795*
  intercept shift add factor *803*
  linked equations *795*
  missing value handling *815*
  Newton solution *960*
  overriding variables *779*, *802*, *805*
  properties of equations *797*
  roundoff of solution *816*
  scenarios *792*, *800*
  simultaneous and recursive blocks *799*
  solution methods *814*
  solving *804*
  solving to match target *816*
  starting values *815*
  static solution *809*
  stochastic simulation *808*
  stochastic solution *810*
  text description of *799*
  text keywords *799*
  tracking variables *812*
  updating links *796*
  variable dependencies *798*
  variable shift add factor *803*
  variable view *798*
Mouse *18*
Move window *20*
Moving average
  geometric *144*
Multicollinearity *460*

N

NA *135*
  inequality comparison *135*
  test *136*
NA. See also Missing data.
Nadaraya-Watson *405*
Name
  object *81*
  reserved *81*
Name field in database query *281*
Near singular matrix *460*
  binary models *628*
  nonlinear models *487*, *953*
  polynomial distributed lag *462*
  RESET test *588*
Nearest neighbor fit *402*
Negative binomial model *659*
Newey-West
  HAC covariance *473*
  truncation lag *473*
Newton's method *814*, *960*
Newton-Raphson *957*
Noninvertible MA process *507*, *512*
Nonlinear coefficient restriction
  Wald test *575*
Nonlinear least squares *480*
  convergence criterion *483*
  forecast standard errors *552*
  iteration option *483*
  specification *481*
  starting values *482*
  two stage *485*
  two stage with AR specification *486*
  weighted *484*
  weighted two stage *486*
  with AR specification *485*, *498*
Nonnested tests *595*
Nonstationary time series *517*
Normal distribution
  test for *323*, *580*
Normalize formula *140*
N-step forecast test *591*
Null hypothesis *569*
Numbers
  relational comparison *132*
N-way table *385*
  chi-square tests *383*

O

Object *73*
  allow multiple untitled *938*
  basics *74*
  closing untitled *938*
  copy *82*
  create *77*
  data *74*
  delete *84*
  freeze *84*
  icon *75*
  label *82*
  naming *82*
  open *78*
  procedure *75*
  sample *104*
  show *78*
  store *85*
  type *75*
  window *78*
Objects menu *81*
Observation equation *754*
ODBC *54*
OLS (ordinary least squares)
  adjusted R-squared *451*
  coefficient standard error *450*
  coefficient t-statistic *450*
  coefficients *449*
  standard error of regression *452*
  sum of squared residuals *452*
  system estimation *696*, *713*
Omitted variables test *577*, *587*
One-step forecast test *591*
One-way tabulation *325*
Open
  database *263*
  multiple objects *77*
  object *77*
  options *259*
  workfile *63*
Operator *129*
  arithmetic *129*
  conjunction (and, or) *133*
  difference *134*
  lag *133*

lead 133
parentheses 130
relational 132
Optimization algorithms
BHHH 958
convergence criterion 953
first derivative methods 957
Gauss-Newton 958
Goldfeld-Quandt 957
grid search 959
iteration control 953
Marquardt 958
Newton-Raphson 957
second derivative methods 956
starting values 951
step size 959
Option setting
frequency conversion 939
warn on close 938
Option settings
allow only one untitled 938
backup workfiles 941
date notation 939
default fonts 937
fonts 937
keyboard focus 938
print setup 943
program execution mode 939
series auto label 943
spreadsheet view defaults 942
warn on close 938
Or operator 98, 133
Order condition for identification 475
Order of stacked data 834
Ordered dependent variable 638
error messages 642
log likelihood 639
Ordinary residual
binary models 634
censored models 650
count models 663
truncated models 656
Orthogonality condition 488, 717
Overdispersion 660, 668
specification test 663
Overidentifying restrictions 491

P

Pack database 288
Packable space 264, 288
Panel
cell identifier 879
dynamic panel data 905
GMM estimation 905
group identifier 878
instrumental variables estimation 904
least squares estimation 901
samples in panel workfiles 881
within-group identifier 880
Panel data
balanced 214
convert to pool 244
dated 214
duplicate identifiers 212, 228
identifiers 210
irregular 214
lags and leads 881
lags in 212
nested 217
regular 214
time trend 885
unbalanced 214
undated 214
unit root tests 530
Param (command) 483, 703, 952
Parks estimator 866
Partial autocorrelation 502
function 327
Paste 83
existing series 109
new series 108
PcGive data 293
PDL (polynomial distributed lag)
forecast standard errors 552
instruments 463
near end restriction 462
specification 462
Phillips-Perron test 524
Pie graph 376
Poisson model 659
Polynomial distributed lag
far end restriction 462
Polynomial distributed lags 552

Pool
and cross-section specific series 829
AR specification 847
balanced data 835, 840
balanced sample 846
coefficient test 859
common coefficients 847
cross section identifiers 828
cross-section specific coefficients 847
defining groups of identifiers 828
dummy variable 838
editing definitions 828
export data 844
fixed effects 863
generate series 838
identifier comparison with "?" wildcard 948
import data 832
import stacked data 835
instrumental variables 851, 867
make group 842
make system 843
naming series 829
order 834
period-specific coefficients 847
pool series 830
random effects 863
residuals 859
special group identity series 830
stacked data 833
unstacked data 832
workfile 825
Pool (object)
? placeholder 830
base name 829
convergence criterion 850
copy 828
Pooled data 825
convert to panel 251
PostScript
save graph as PostScript file 429
Prais-Winsten 500
Precedence of evaluation 130
Predetermined variable 473
Prediction table
binary models 629
ordered models 642
Prewhitening 705, 720
Principal components 385
Print
setup options 943
Probability response curve 635
Probit models 623
Procedure 75
Program
backup files 939
execution option 939
p-value 569
for coefficient t-statistic 450

Q

QML 660
QQ-plot (quantile-quantile) 323, 393
multiple 378
Q-statistic
Ljung-Box 328
residual serial correlation test 726
serial correlation test 495
Quadratic hill-climbing 957
Quadratic spectral kernel 718
Qualitative dependent variable 621
Quantile 392
Quasi-generalized pseudo-maximum likelihood 665
Quasi-likelihood ratio test 662, 666
Quasi-maximum likelihood 660
robust standard errors 667
Queries on database 277
advanced query 280
DRI 304
easy query 278
examples 284
logical expressions 279
wildcard characters 278
Quiet mode 939

R

Ramsey RESET test 586
Random effects 863
Random effects test 928
Random walk 517
Randomize ties 632
Rank condition for identification 475
Ratio to moving-average 350

RATS data
4.x native format 294
portable format 294
Read 832
Rebuild database 289
Recursive coefficient 592
save as series 592
Recursive estimation
least squares 588
using state space 763
Recursive residual 588, 589
CUSUM 589
CUSUM of squares 590
n-step forecast test 591
one-step forecast test 591
save as series 592
Redundant variables test 579
Registry 275
Regression
adjusted R-squared 451
coefficient standard error 450
collinearity 460
forecast 543
F-statistic 453
log likelihood 452
residuals from 457
standard error of 452
sum of squared residuals 452
t-statistic for coefficient 450
Relational operators
and missing values 135
Remarks
field in database query 284
Rename 81
database 287
objects in database 271
workfile page 70
Repair database 289
Representation view
of equation 455
Reserved names 81
RESET test 586
Residuals
default series RESID 457
from estimated equation 457
from two stage least squares 476
generalized 634, 650, 656, 663
make series 456
of equation 455
ordinary 634, 650, 656, 663
recursive 588, 589
standardized 456, 634, 650, 656, 663
sum of squares 452
symmetrically trimmed 652
tests of 579
unconditional 494, 499
Resize
workfile 219
Resize window 20
Restore window 19
Restricted estimation 447
Restricted log likelihood 626
Results
accessing from estimated equation 454
Rich Text Format 437
RichText Format
save table as Rich Text file 438
Robust standard errors
Bollerslev-Wooldridge for GARCH 607
GLM 667
Huber/White (QML) 667
Robustness iterations
with nearest neighbor fit 404
with regression 402
Root mean squared error 553
Row height 433
R-squared
adjusted 451
for regression 451
from two stage least squares 476
McFadden 626
negative 609
uncentered 581, 583
with AR specification 500
RTF 437, 438

S

Sample
@all 97
@first 97
adjustment in estimation 448
balanced 846
change 96
command 98

common 135
current 58
if condition 97
individual 135
selection and missing values 98
specifying sample object 104
specifying samples in panel workfiles 881
used in estimation 448
using sample objects in expressions 104
with expressions 99
workfile 95
SAR specification 506
SAS file 54
Save
backup workfile 62
options 259
save as new workfile 62
workfile 61
workfile precision and compression 62
Scale factor 649
Scatter diagrams 374, 400
matrix 377
multiple 376
simple 374, 400
with kernel fit 405
with nearest neighbor fit 402
with regression 374, 400, 401
Scatterplot 375
Scenarios in models 792
Schwarz criterion 971
for equation 453
Score vector 635
Scroll window 18
Seasonal
ARMA terms 506
difference 134, 504
Seasonal adjustment
additive 350
Census X11 (historical) 344
Census X12 337
multiplicative 350
Tramo/Seats 345
Seasonal graphs 310
Second derivative methods 956
Seemingly unrelated regression 697
Select
all 77
multiple items 20
object 77
single item 20
Sensitivity of binary prediction 630
Serial correlation
ARIMA models 501
Durbin-Watson statistic 452, 494
first order 494
higher order 494
nonlinear models 498
tests 494, 581
theory 493
two stage regression 499
Series
area graph view 310
auto-updating 149
auto-updating and databases 152
auto-updating and forecasting 561
bar graph view 310
create 88, 137
cross-section specific 829
delete observation 93
descriptive statistics 310
difference 134
edit in differences 364
editing 91
in pool objects 830
insert observation 93
label 333
lag 133
lead 133
line graph view 310
properties 332
seasonal graph view 310
smpl + /- 91
spike graph view 310
spreadsheet view 309
spreadsheet view defaults 942
Shade region of graph 423
Shadowing of object names 286
Sharing violation 265
Show 78
Siegel-Tukey test 322
Sign test 317
Simple hypothesis tests 315
mean 315
median 317
variance 316

Simple scatter plot *374*, *400*
Skewness *311*
SMA specification *506*
Smoothing
  methods *351*
  parameters *351*
Smpl command *98*
Smpl + /- *91*
Solve
  Gauss-Seidel *959*
Source
  field in database query *284*
Sparse label option *313*, *382*
Specification test
  for binary models *637*
  for overdispersion *663*
  for tobit *652*
  RESET (Ramsey) *586*
  White *583*
Specificity of binary prediction *630*
Specify equation *444*, *569*
  in systems *699*
  nonlinear *481*
Spike graph *310*
Spreadsheet
  file export *115*
  file import *110*
  view option *942*
SPSS file *54*
Stability test *584*
  Chow breakpoint *584*
  Chow forecast *585*
  RESET *586*
  with unequal variance *593*
Stacked data *833*
  balanced *835*
  descriptive statistics *840*
  order *834*
Standard deviation *311*
Standard error
  for estimated coefficient *450*
  forecast *552*, *564*
  of the regression *452*
Standard error
  See also Robust standard errors. *450*
Standardized residual *456*
  binary models *634*
  censored models *650*
  count models *663*
  truncated models *656*
Start
  field in database query *283*
Starting values
  (G)ARCH models *606*
  binary models *627*
  for ARMA estimation *509*
  for coefficients *483*
  for nonlinear least squares *482*
  for systems *703*
  param statement *483*, *952*
  user supplied *510*
Starting values for coefficients *951*
Stata file *54*
State equation *754*
State space *753*
  @mprior *763*
  @vprior *763*
  estimation *767*
  observation equation *754*
  representation *753*
  state equation *754*
State variables *753*
Static forecast *557*
Stationary time series *517*
Status line *24*
Step size *959*
Store *85*
  as .DB? file *267*
  from pool *844*
  in database *267*
  merge objects *268*
Structural change *584*
Structural forecast *557*
Structural solution of models *809*
Structural VAR *733*
  estimation *738*
  factorization matrix *727*
  identification *737*
  long-run restrictions *735*
  short-run restrictions *734*
Sum of squared residuals
  for regression *452*
Summary statistics
  for regression variables *451*

SUR *697*, *715*
Survivor function *392*
Symmetrically trimmed residuals *652*
System
  create *698*
  definition *695*
  estimation *696*
  estimation methods (technical) *712*
  specification *699*

T

Table *429*
  cell formatting *433*
  column width *433*
  copy to other windows programs *437*
  edit *432*
  font *434*
  paste as unformatted text *437*
  save to file *438*
Tables
  colors *434*
  fonts *434*
  selecting cells *430*
Tabulation
  n-way *381*
  one-way *325*
Template *424*
Tests. See also Hypothesis tests, Specification test and Goodness of fit.
Text (object) *438*
Text file
  open as workfile *54*
Theil inequality coefficient *554*
Three stage least squares *697*, *716*
Threshold GARCH (TARCH) *613*
Title bar *22*, *58*, *79*
to (lag range) *445*
Tobit *645*
Toggling *21*
Toolbar *58*, *79*
Tracking model variables *812*
Tramo/Seats *345*
Transition equation *754*
Transpose *363*
Truncated dependent variable *654*
  fitted index *656*
  log likelihood *654*
Truncation point *655*
TSD data format *293*
TSP portable data format *295*
Two stage least squares *480*
  in systems *697*
  nonlinear *485*
  nonlinear with AR specification *486*
  order condition *475*
  rank condition *475*
  residuals *476*
  system estimation *715*
  weighted *476*
  weighted in systems *697*, *715*
  weighted nonlinear *486*
  with AR specification *477*, *499*
  with MA specification *479*
Type
  field in database query *282*

U

Unconditional residual *499*
Unit root test *329*, *518*
  augmented Dickey-Fuller *522*
  Dickey-Fuller *522*
  Dickey-Fuller GLS detrended *523*
  Elliot, Rothenberg, and Stock *525*
  KPSS *525*
  panel data *530*
  Phillips-Perron *524*, *525*
  pooled data *841*
  trend assumption *523*
Units
  field in database query *284*
Unstacked data *832*
Untitled *81*, *82*
Update
  coefficient vector *457*, *952*
  group *363*
Urzua factorization matrix *727*
User supplied starting values *510*

V

Valmap
  find label for value *173*
  find numeric value for label *174*

find string value for label *174*
functions *173*
Value maps *163*
Van der Waerden test *317*, *321*
VAR
AR roots *724*
autocorrelation LM test *726*
decomposition *731*
estimation output *723*
factorization matrix in normality test *726*
Granger causality test *724*
impulse response *729*
Jarque-Bera normality test *726*
lag exclusion test *725*
lag length choice *725*
mathematical model *721*
See also Impulse response.
See also Structural VAR.
Variance decomposition *731*
Variance factor *668*
Variance proportion *554*
VEC *749*
estimating *749*
Vector autoregression *721*
Verbose mode *939*
Version 5 compatibility *4*
View
default *78*
Volatility *602*

W

Wald test *572*
coefficient restriction *572*
formula *576*
F-statistic *577*
joint restriction *574*
nonlinear restriction *575*
structural change with unequal variance *593*
Warning on close option *938*
Watson test *323*
Weighted least squares *471*
cross-equation weighting *696*
nonlinear *484*
nonlinear two stage *486*
pool *848*
system estimation *714*
two stage *476*
two stage in systems *697*, *715*
Weighting matrix
for GMM *488*, *717*
heteroskedasticity and autocorrelation consistent (HAC) *718*
kernel options *718*
White heteroskedasticity consistent *717*
White
heteroskedasticity consistent covariance *472*
Width of table column *433*
Wilcoxon signed ranks test *317*, *320*
Wildcard characters *59*, *945*
in easy query *278*
Window
active *18*, *79*
database *263*
EViews main *21*
object *81*
scrolling *18*
size control *20*
Within deviations *886*, *893*
Work area *24*
Workfile
automatic backup *941*
directory *58*
load existing from disk *63*
multi-page *64*
remove structure *231*
resize *231*
resizing *219*
sample *95*
save *61*
stacking *251*
storage defaults *940*
storage precision and compression *941*
summary view *61*
unstacking *244*
window *57*
Write *844*

X

X11 (historical) *344*
limitations *345*
X11 using X12 *338*
X12 *337*
XY line
view *375*, *378*

Y

Yates' continuity correction *321*